(1)

AFIT/GAE/ENY/92D-23

# THE EFFECTS OF VISCOSITY ON A CONICALLY DERIVED WAVERIDER

THESIS

James A. Mundy V
First Lieutenant, USAF

AFIT/GAE/ENY/92D-23

Approved for public release, distribution unlimited

AFIT/GAE/ENY/92D-23

# THE EFFECTS OF VISCOSITY ON A CONICALLY DERIVED WAVERIDER

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Aeronautical Engineering

James A. Mundy V, B.A.E

First Lieutenant, USAF

December 1992

Approved for public release; distribution unlimited

## Acknowledgments

I owe much to many for the patience and help they extended to me in the completion of this work. I am particularly grateful to Dr. Datta Gaitonde, whose hard work convinced the computer code to run properly, and who always took time to answer my questions, often putting his own work aside to do so. My thanks for his many hours of work debugging the code and helping me to get the solutions to converge. I am also grateful to my adviser, Lt Col Gerald Hasen. Lt Col Hasen's aid in getting the necessary computer resources was invaluable, as were his suggestions and comments throughout the whole process of research and writing. His guidance, and cracking of the proverbial whip, kept me on track and moving. I must also extend a thank you to Capt Gregory Stecklein. Capt Stecklein took the time to provide me with his computer code, and explanations, and so saved me many headaches. His excellent work gave me a good start for my own efforts. Ohio Supercomputer Center also has my thanks, both for the initial grant they extended to me, and for providing an extension of the grant when it was necessary.

My thanks for the friendship and support of my fellow students goes to them, along with my wish for their success in whatever they set their hands to. My wife, Susan, has my particular gratitude. Her patience with a highly distracted husband, and constant encouragement, is deserving of more than I can say. It is to her that this work is dedicated. Most of all though, my thanks go to God. Only His gifts have enabled me to accomplish as much as I have.

James A. Mundy V

## TABLE OF CONTENTS

## List of Figures

## List of Tables

# List of Symbols

| | |
|---|---|
| $\mathbf{A}$ | Flux Vector Jacobian Matrix |
| $\mathbf{B}$ | Flux Vector Jacobian Matrix |
| $\mathbf{C}$ | Flux Vector Jacobian Matrix |
| $C_D$ | Coefficient of Drag |
| $C_{Dw}$ | Coefficient of Wave, or pressure, Drag |
| $C_{Dv}$ | Coefficient of Viscous Drag |
| $c_f$ | Coefficient of Skin Friction |
| $C_L$ | Coefficient of Lift |
| $c_p$ | Coefficient of Pressure |
| D | Total Drag, including pressure and viscous components |
| D | Difference Operator: $Df = f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}}$ |
| $D_w$ | Wave Drag, drag due to pressure |
| $D_v$ | Viscous Drag, drag due to viscous effects |
| $E_t$ | Total Energy |
| $\mathbf{E}$ | Flux Vector |
| $\mathbf{F}$ | Flux Vector |
| $\mathbf{G}$ | Flux Vector |
| f | Similarity Momentum Function |
| g | Y-Coordinate of Spline Endpoints |
| $\mathbf{J}$ | Jacobian Matrix |
| J | Total Enthalpy |

| | |
|---|---|
| $L$ | Total Lift |
| $m$ | Slope of a Line |
| $M_{\int}$ | Freestream Mach Number |
| $p_{\int}$ | Freestream Pressure at a Standard Altitude |
| Pr | Prandtl Number |
| $q_{\int}$ | Freestream Dynamic Pressure |
| $r$ | Radial distance from the axis of the generating cone to an arbitrary point in the flow |
| $R_o$ | Non-Dimensional Offset Distance, taken from the generating cone's axis to the waverider's plane of symmetry |
| $R_{\infty b}(\phi)$ | Polar Mapping of the Waverider Freestream Surface |
| $R_{cb}(\phi)$ | Polar Mapping of the Waverider Compression Surface |
| $r_s(\phi)$ | Polar Mapping Function of the Leading Edge of the Waverider |
| Re | Reynolds Number |
| $s$ | Slope in Spline Calculation |
| $S$ | Similarity Enthalpy Function |
| $S_p$ | Waverider Planform Area |
| $S_w$ | Waverider Wetted Area |
| $St^*$ | Stanton Number |
| $u$ | Cartesian Coordinate X-component of Velocity |
| $v$ | Cartesian Coordinate Y-component of Velocity |
| $w$ | Cartesian Coordinate Z-component of Velocity |
| $U$ | Density, Momentum, and Energy Vector |

| | |
|---|---|
| $V_r$ | Radial Component of Velocity |

**Greek**

| | |
|---|---|
| $\alpha$ | Angle of Attack |
| $\beta$ | Shock Angle |
| $\chi$ | Hypersonic Viscous Interaction Parameter |
| $\delta$ | Generating Cone Half-Angle |
| $\delta$ | Difference Operator: $\delta f = f^{n+1} - f^n$ |
| $\delta^*$ | Displacement Thickness |
| $\gamma$ | Ratio of Specific Heats, assumed constant $\gamma = 1.4$ |
| $\lambda$ | Second Coefficient of Viscosity |
| $\lambda_i$ | Eigenvalue |
| $\Lambda$ | Eigenvalue Matrix |
| $\mu$ | Coefficient of Viscosity |
| $\phi$ | Dihedral Angle, range from $-\phi_l \leq \phi \leq \phi_l$ |
| $\phi_l$ | Dihedral Angle of Waverider at Baseplane |
| $\phi_x$ | Dihedral Angle of Waverider at an Arbitrary Cross Section |
| $\Pi$ | Enthalpy Thickness |
| $\theta$ | Arbitrary Angle, range from $\delta \leq \theta \leq \beta$ |
| $\theta$ | Momentum Thickness |
| $\rho$ | Density |
| $\sigma$ | Ratio of $\beta$ to $\delta$ |
| $\tau$ | Z-Coordinate of Spline Endpoints |

| | |
|---|---|
| $\tau_{ij}$ | Shear Stress |
| $\xi$ | Computational Streamwise Axis |
| $\eta$ | Computational Normal Axis |
| $\zeta$ | Computational Spanwise Axis |
| $\eta$ | Similarity Normal Axis |
| $\xi$ | Similarity Streamwise Axis |

AFIT/GAE/ENY/92D-23

## Abstract

This study investigated the effects of the interaction between the viscous boundary layer and the shock wave produced by a Mach 10 inviscid optimized waverider. An implicit, Roe flux-splitting algorithm, developed by WL/FIMM, was used to solve the flow field. A validation for the inviscid version of the CFD algorithm was accomplished by comparing the numerical data produced by the CFD code to the analytic results derived by Rasmussen, and by comparison to results of the explicit version of the same Roe flux-splitting code. The computational results compared favorably. The inviscid case studied using the implicit code produced results identical, for all practical purposes, to those of the explicit code, though approximately twice as quickly. The results of the viscous flow case matched well with the results predicted by theory. The lift to drag ratio calculated, 5.74, is comparable to the results of other researchers.

# I. INTRODUCTION

## 1.1 Background

The waverider concept was introduced in 1958 by Nonweiler and Hilton (24), with an expanded discussion subsequently published by Nonweiler in 1959. The three-dimensional body Hilton and Nonweiler examined was deemed the "caret wing" because of the distinctive caret shape of its base plane (see Figure 1-1).

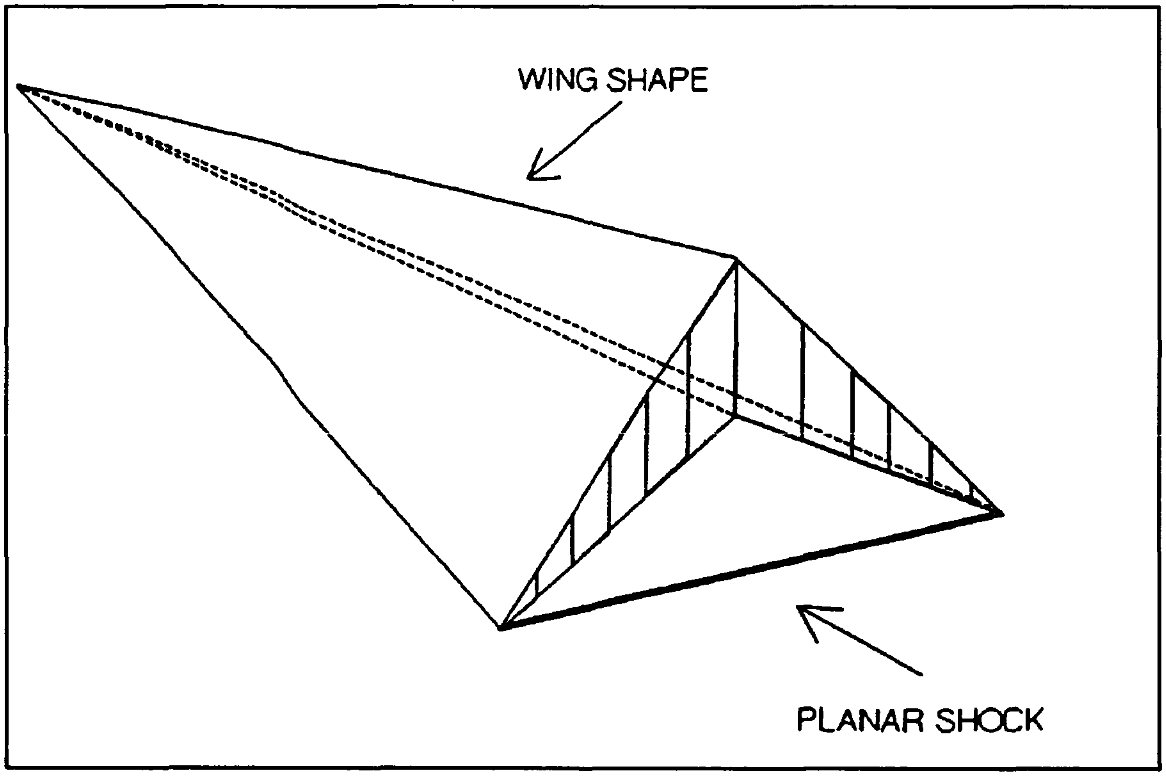

**Figure 1-1**, Caret Wing Waverider

A waverider is a lifting body which is derived from a known, analytic flow field, such as flow over a two-dimensional wedge or flow around a slender cone. The body is designed so that the shock is attached along the waverider's entire leading edge, thus capturing the high pressure region behind the shock entirely on the bottom surface of the waverider. The top surface of the vehicle is typically designed to be completely parallel to the freestream, thus creating no wave drag or pressure drag. The substantial difference in pressure between the freestream (upper) and compression (lower) surfaces of the waverider generates lift. This maximizes the lift to drag (L/D) ratio, one of the primary goals of high speed aircraft designs.

Hilton's and Nonwieler's initial waverider work was quickly followed up by others, with the efforts concentrating on two-dimensional generating flow fields, such as flow over wedges. Rasmussen (26) extended the waverider concept to axisymmetric flow fields generated by slender conical bodies using Hypersonic Small Disturbance Theory (HSDT) to analyze flow around the generating conical bodies and to develop simplified equations which describe the shape of the waverider based upon the generating flow field. Rasmussen's work in the 1980's concentrated primarily on optimizing waveriders for specific flight conditions, with Mach number and generating cone angle being the dominant parameters.

Thus far, all the work mentioned has been inviscid, i.e., the effects of viscosity were neglected. Waveriders tend to suffer from viscous effects, as they have a comparatively low volume and a very large wetted surface area; thus much research has been focused on examining viscous flow over waveriders. Bowcutt, et al. (10, and 11), examined viscous optimized waveriders, optimizing for maximum L/D, by means of integral boundary layer methods. Corda (12) also investigated viscous optimized waveriders derived from a flow field generated by a power-law minimum drag body, where a reference temperature method was used to calculate the skin friction. At high Mach numbers, Corda's method gives higher L/D values than Bowcutt's, although Bowcutt's method does better than Corda's at Mach numbers less than 10. McLaughlin (21) has investigated the use of chemically reacting conical flow for the generating flow field. Vanmol (33) optimized his waverider shape for maximum L/D using aerodynamic heating as a constraint. Chang (6) investigated the effect of viscous boundary layer / shock interaction on waverider optimization through the use of the hypersonic interaction parameter $\chi$ (defined in Chapter 5). Takashima (31) validated the integral boundary layer method, using a full Navier-Stokes solver, by investigating a viscous optimized waverider at Mach 6.

## 1.2 Objectives

There are several reasons for performing this thesis research. The first of these reasons is that the determination of viscous effects on waveriders has by no means been fully examined. For instance, the sharp leading edges and nose common to inviscid waveriders do not take into account the extreme aerodynamic heating such vehicles will encounter. While Takashima (31) investigated rounded leading edges, it was only for one variety of waverider over a narrow range of flight conditions. This thesis will significantly contribute to the database of computational solutions to waverider designs by examining conically derived waveriders with both sharp and blunted leading edges, including a detailed examination of viscous interaction effects, allowing a direct comparison between the inversely designed, analytic, inviscid flow field, and the numerically computed inviscid and viscous flow fields. Another primary reason for this thesis is to provide validation results for the Wright Laboratory's high speed, implicit, finite-volume Navier-Stokes code for a vehicle of comparable shape to the forebodies that have been considered for the National Aerospace Plane (NASP). This code has, to date, been run on only one three-dimensional configuration, the X-24C. Specific objectives this thesis will accomplish are:

(1) Develop the body coordinates of an inviscidly derived hypersonic waverider, and modify it to round the leading edges. The edges will be rounded by separating the freestream surface from the compression surface by an appropriate amount, and then fitting a curve between the edge points to provide a smooth leading edge.

(2) Generate a three-dimensional grid suitable for capturing the expected shock structure and the boundary layer on the body.

(3) Apply both the Euler and the full Navier-Stokes versions of the Wright Laboratory's three-dimensional, implicit, Roe flux difference splitting flow solver developed by Gaitonde (15) to the hypersonic waverider and grid developed in paragraphs (1) and (2) above.

(4) Compare the inviscid computational results to inviscid analytical results (i.e., Rasmussen (25) ) for similar configurations, and then examine the effects of viscosity by comparing the viscous and inviscid computational solutions.

## 1.3 Methodology

Rasmussen (25) and (26), details the waverider design methodology applied herein. The process is essentially an inverse design, where the waverider shape is derived from streamlines in a previously known flowfield. The vehicle investigated in this effort is based on axisymmetric hypersonic flow past a slender, right circular cone. A Cartesian coordinate system, with the X-axis along the cone center, the Y-axis pointing downward, and the Z-axis in the spanwise direction is used. Figure 1-2 illustrates the Cartesian coordinate system used in the analysis.

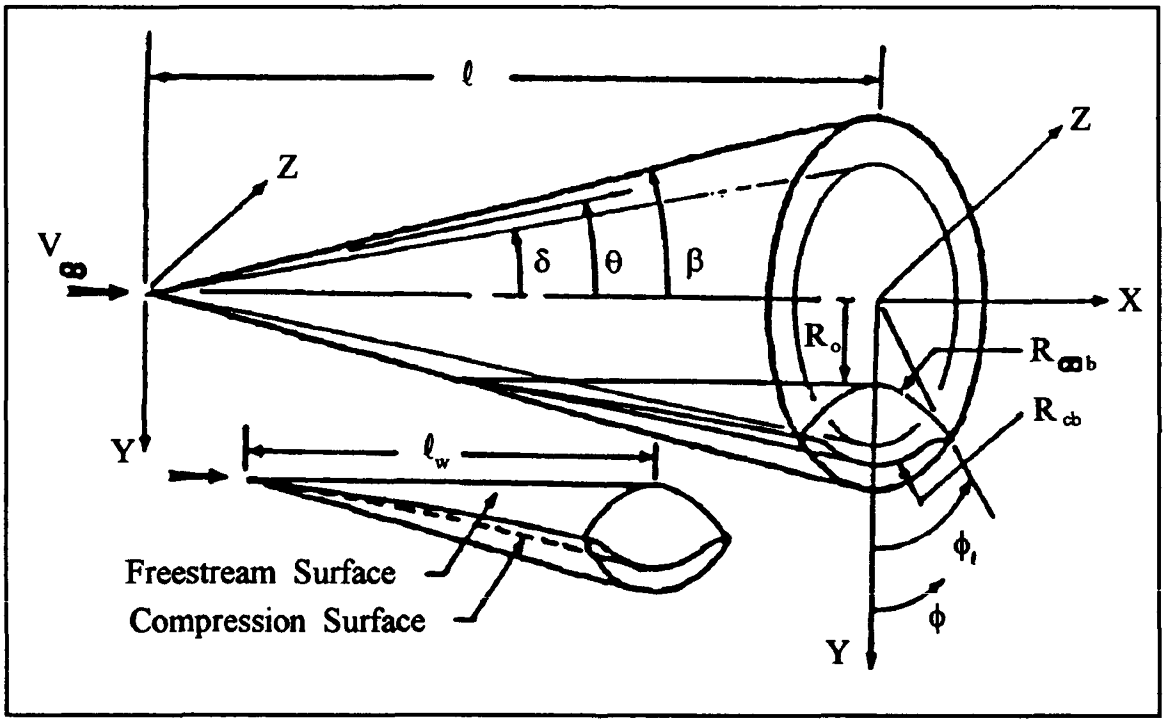

**Figure 1-2, Waverider, Generating Cone & Coordinate System**

Since Hypersonic Small Disturbance Theory (HSDT) is more amenable to spherical coordinates, the Cartesian system described above was transformed to a spherical coordinate system with the waverider's nose as the origin, the angle $\theta$ measured from the X-axis, and $\phi$, the azimuthal angle measured from the Y-axis in the Y-Z plane. The basic parameters that determine the conical flow geometry are $\delta$, the cone semi-angle, $\beta$, the shock angle, and $\sigma$ ($\beta/\delta$), the ratio of shock angle to cone semi-angle. Figure 1-2 illustrates the spherical coordinates used herein.

As mentioned previously, the waverider is derived using HSDT. The process is given in some detail in chapter 2, but the basic method used to derive the waverider is as follows:

1. The trailing edge of the freestream surface is defined with a four-term, sixth-order polynomial given by Rasmussen (26), Eqn (2-4). This effort will use a parabolic top waverider, which uses only the first two terms of the polynomial.

2. The flowfield is solved analytically. For hypersonic flow around a slender cone, an analytic solution to the Taylor-MacColl equation (7), is obtained using the HSDT technique.

3. Now that the generating flow field is analytically defined, streamlines parallel to the freestream are traced upstream from the freestream trailing edge until they intersect the shock wave produced by the generating cone, as shown in Figure 1-3. This defines the entire freestream surface and the leading edge of the waverider.

4. From the leading edge defined in step 3, streamlines are traced downstream, using the HSDT relations, until they cross the chosen baseplane, as shown in Figure 1-3. These streamlines define the waverider's compression surface.

Now that the waverider body is completely defined, a computational grid is generated to define the domain in which the CFD algorithm will be applied. The GRIDGEN package (30) was used to generate the grid. Ideally, the grid will scale with the waverider in the streamwise direction, so that the shock wave will be captured at

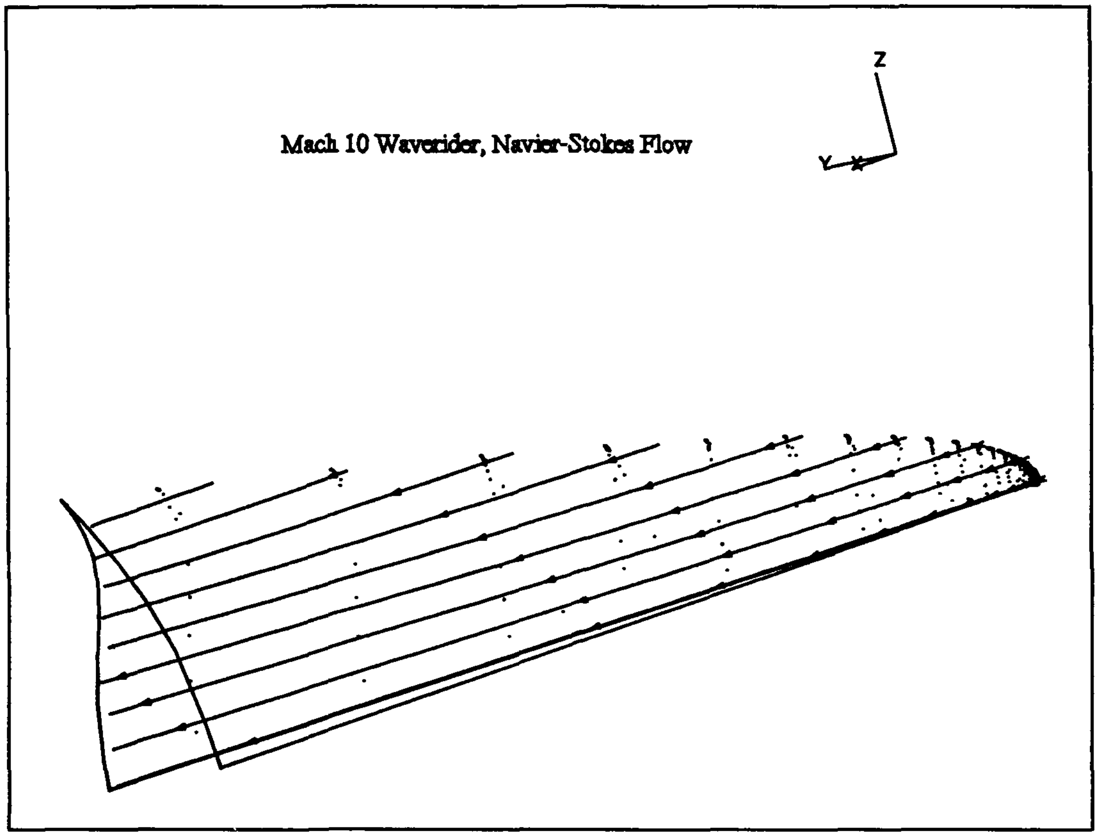

**Figure 1-3**, Streamlines on the Waverider's Surface

approximately the same grid location for each cross section. The scaling also preserves computational resources, which would otherwise be wasted by calculating regions of the flow which should remain at freestream conditions. However, due to viscous effects, it may not be possible to maintain the shock at the same grid location, as viscous hypersonic interaction will cause the shock to have a steeper angle than predicted by inviscid flow theory. This is particularly true close to the waverider's nose where boundary layer growth is rapid, and the shock/boundary layer interaction is strong.

The CFD algorithm used to obtain a computational flow solution is an implicit, flux splitting Roe-averaged, finite-volume scheme. The Roe-averaging scheme is particularly desirable in this case due to its shock capturing capability and stability at high Mach numbers. The algorithm, as implemented by Gaitonde (16), is also quite robust relative to grid skewness and accurate to second order in both space and time. Numerical values of the velocity components $(u, v, w)$, Mach number, pressure coefficient, lift, wave drag and viscous drag obtained from the CFD code will be compared to analytic data (25), and other numerical solutions.

# II. WAVERIDER SURFACE FORMULATION AND GRID GENERATION

## 2.1 Waverider Surface Formulation

The equations defining a given waverider body are derived in detail in Appendix A. Briefly, the waverider is described by the following equations, with the coordinate systems illustrated by Figure 1-2. Except where noted, the following equations are derived by Rasmussen (26). To begin, the freestream surface is described by:

$$r\theta = r_s(\phi)\beta \tag{2-1}$$

and the compression surface by:

$$r(\theta^2 - \delta^2)^{\frac{1}{2}} = r_s(\phi)(\beta^2 - \delta^2)^{\frac{1}{2}} \tag{2-2}$$

Equations (2-1) and (2-2) are given in spherical coordinates, with the nose of the generating cone being at the origin. Note that $r = r_s(\phi)$ is the leading edge of the waverider, and $\phi$ is the included angle measured from the line of symmetry to the leading edge. Note also that the leading edge is also the intersection of the waverider body with the shock.

The relation of the cone half angle, $\delta$, to the shock angle, $\beta$, is provided by HSDT as

$$\sigma = \frac{\beta}{\delta} = \left(\frac{\gamma + 1}{2} + \frac{1}{M_\infty^2 \delta^2}\right)^{\frac{1}{2}} \tag{2-3}$$

The freestream surface of the waverider is defined by the four term, sixth order polynomial

$$Y = R_o + AZ^2 + BZ^4 + CZ^6 \qquad \textbf{(2-4)}$$

where $R_o$, $A$, $B$, and $C$ are constants which determine the curvature of the freestream surface of the waverider. The polynomial given by Eqn (2-4) must satisfy two conditions

$$Y = \begin{cases} R_o & : \; Z = 0 \\ \sigma \cos\phi_l & : \; Z = \sigma \sin\phi_l \end{cases} \qquad \textbf{(2-5)}$$

where the angle $\phi_l$ is the included angle at the baseplane.

At the baseplane, the compression surface, as derived by Rasmussen (26) is defined by:

$$R_{cb}^2(\phi) = 1 + \left(\frac{\sigma^2 - 1}{\sigma^2}\right) R_{\infty b}^2(\phi) \qquad \textbf{(2-6)}$$

The same form of Eqn (2-6) holds for the entire length of the waverider. The included angle, $\phi$, however, decreases with decreasing $X$. The shock relation parameter, $\sigma$, scales such that $\phi_o = 0$ at the nose coincides with $\sigma_o = R_o$. The shock attachment condition, i.e., the requirement that the shock must attach at the leading edge of the waverider, must still be met at each Y-Z cross section, and is enforced by

$$\begin{gathered} Y = \sigma_X \cos\phi_X \qquad Z = \sigma_X \sin\phi_X \\ \sigma_X \cos\phi_X = R_o + A\sigma_X^2 \sin^2\phi_x \end{gathered} \qquad \textbf{(2-7)}$$

An equation for $\sigma_X$ results, since it is known that $\sigma$ ranges from $\sigma_o = R_o$ to $\sigma_l = \sigma$. Thus

$$\sigma_X = R_o + \frac{X_w}{l_w}(\sigma - R_o) \qquad \textbf{(2-8)}$$

where $X_w/l_w$ is the non-dimensionalized length of the waverider.

Finally, from Stecklein (28), $R_{cb}$ at an arbitrary Y-Z cross section is given by

$$R_{cb}^2(\phi) = \left(\frac{X_w}{l_x}\left(1 - \frac{R_o}{\sigma}\right) + \frac{R_o}{\sigma}\right)^2 + \left(\frac{\sigma^2 - 1}{\sigma^2}\right)R_{\infty b}^2(\phi) \qquad \textbf{(2-9)}$$

However, the waverider, as developed by Rasmussen (25) is unsuited to viscous, hypersonic flow due to its very sharp leading edges. Such sharp edges magnify already large heat transfer rates. Stecklein (28) developed a Fortran code to generate a waverider body using the equations developed by Rasmussen (25). In the present effort, Stecklein's Fortran code was modified to generate a waverider suitable for viscous, hypersonic flow as follows:

1. The waverider is translated so that the nose of the waverider is at the origin.
2. The compression surface was then moved 1 cm in the positive Y direction to provide the necessary thickness at the leading edges to blunt them.
3. Using linear interpolation, the X coordinate at which the waverider was 1 cm thick (in Z) was determined, as was the Y coordinate of the centerline of the compression surface. The freestream surface (at centerline), of course, does not vary with X.
4. Splines were fit from the interpolated point to a boundary point imposed .33 cm upstream of the interpolated point, in both the X-Y and X-Z planes. The

.33 cm value was determined through experimenting with various values for the boundary point. The ends of the splines were clamped, i.e., they were required to match the slope of the known portion of the waverider and, at the boundary point, an (arbitrary) large value.

5. Several Y-Z planes were determined, using the splines generated in step 4 as outer bounds, by interpolation of data from the first known plane of the waverider.
6. The main portion of the waverider had splines fit in the Y-Z plane at the leading edges. Again, the splines were clamped and an arbitrary boundary point was imposed. The splines were generated first from the freestream surface to the boundary point, and then from the boundary point around to the compression surface.

In this effort, 28 points were added at each plane using the spline. This number of points gave adequate definition for the model without making the model database file overly large. The boundary point was not actually added to the waverider model as it would have caused a sharp point, the very condition the spline is intended to eliminate.

The spline is given by a cubic polynomial:

$$P_i(x) = C_{1,i} + C_{2,i}(x - \tau_i) + \frac{C_{3,i}(x - \tau_i)^2}{2} + \frac{C_{4,i}(x - \tau_i)^3}{6} \qquad \textbf{(2-10)}$$

where the $C_{j,i}$ are constant for each interval $i$. The coefficients can be solved for fairly easily by using a divided difference table for $P_i$ (see Appendix B).

de Boor (14) gives the coefficients of Eqn (2-10) as

$$
\begin{aligned}
C_{1,i} &= P_i(\tau_i) = g(\tau_i) \\
C_{2,i} &= \frac{dP_i(\tau_i)}{d\tau_i} = P_i'(\tau_i) = s_i \\
C_{3,i} &= \frac{P_i''(\tau_i)}{2} = \frac{[\tau_i,\tau_{i+1}]g - s_i}{\Delta\tau_i} - c_{4,i}\Delta\tau_i \\
C_{4,i} &= \frac{P_i'''(\tau_i)}{6} = \frac{s_i + s_{i+1} - 2[\tau_i,\tau_{i+1}]g}{(\Delta\tau_i)^2}
\end{aligned}
\tag{2-11}
$$

Figure 2-1 and Figure 2-2 show the geometry obtained using the cubic splines at the nose, and Figure 2-3 shows the geometry of the 15th plane from the nose. The 15th plane is depicted, as it is the first plane described by Stecklein's code, with the addition of the cubic spline at the leading edge. Note in Figure 2-1 the flat spot at $Y = 0.0$ caused by leaving the boundary point out of the model definition file.

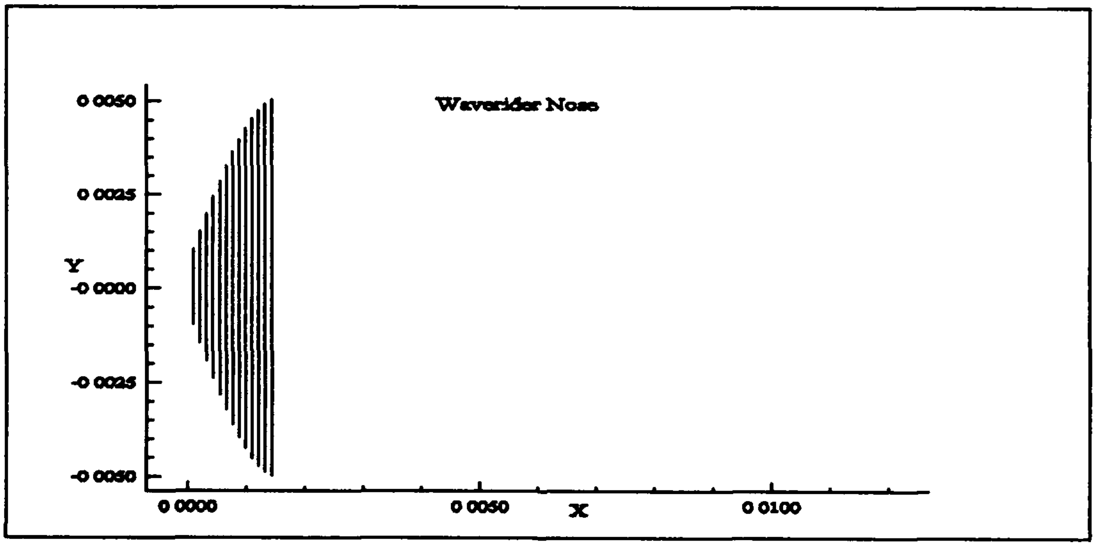

**Figure 2-1**, X-Y Plane of Waverider Nose

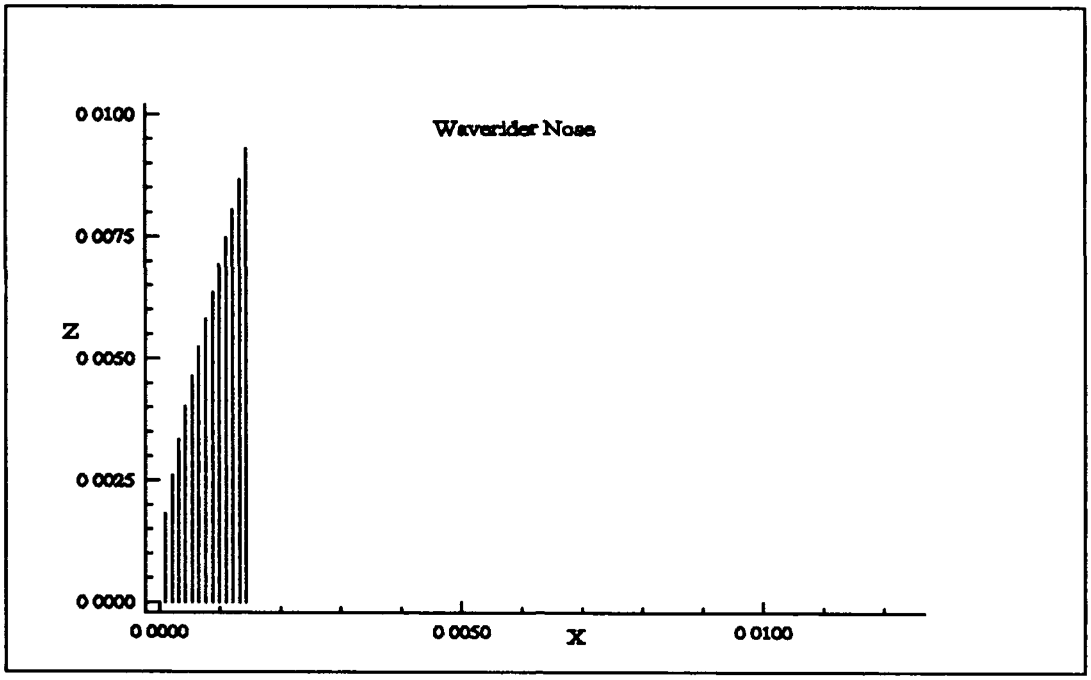

**Figure 2-2**, X-Z Plane of Waverider Nose

Figure 2-4 is a three-dimensional view of the waverider model, showing the results of the whole process described above. The modified Waverider code is in Appendix C.

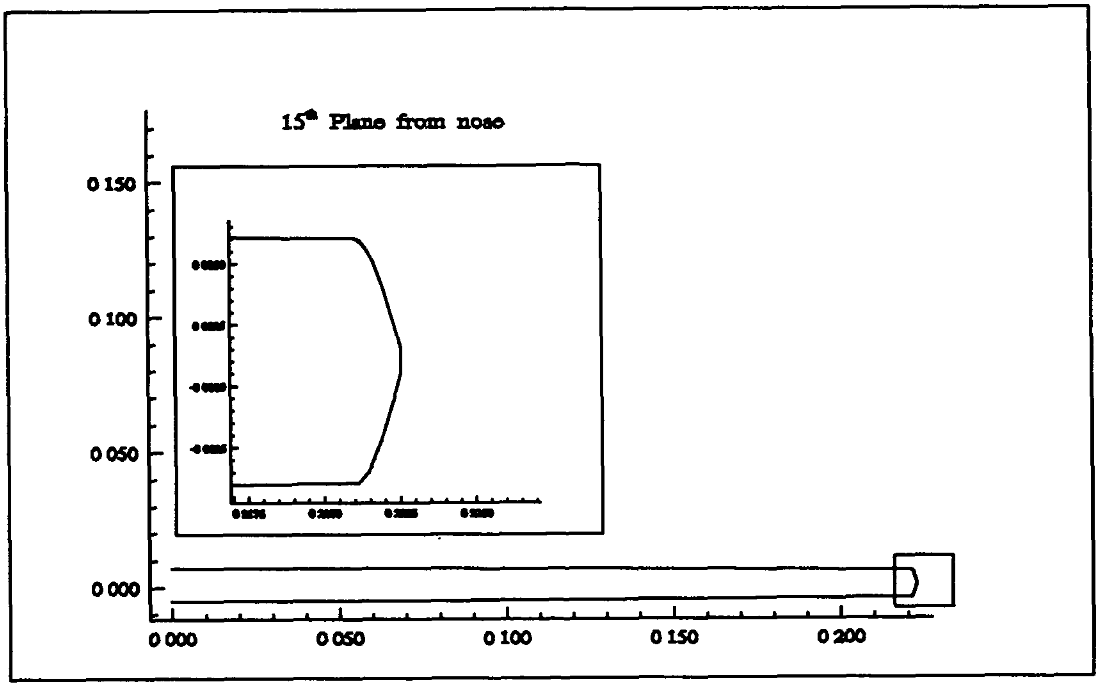

**Figure 2-3,** 15th Plane from Nose

### 2.1.1 Program WAVERIDER

The methods described by Rasmussen (25) and (26), outlined in the previous section were developed into a Fortran program called WAVERIDER by Stecklein (28). Modifications to WAVFRIDER were required to adapt the waverider model the Fortran program produced to one suitable for viscous, hypersonic flow, as WAVERIDER was originally intended to produce a waverider model for an inviscid flow. The modified code defines the surface of a parabolic-top, inviscid optimized waverider, just as written by Stecklein (28), and then adapts it to viscous flow by blunting the leading edges and nose using the methods outlined in the previous section. It must be noted that the code formulates a half-body about the X-Y plane of symmetry. Building the full body would yield profit only if a non-zero yaw was considered (in which case it would be required),

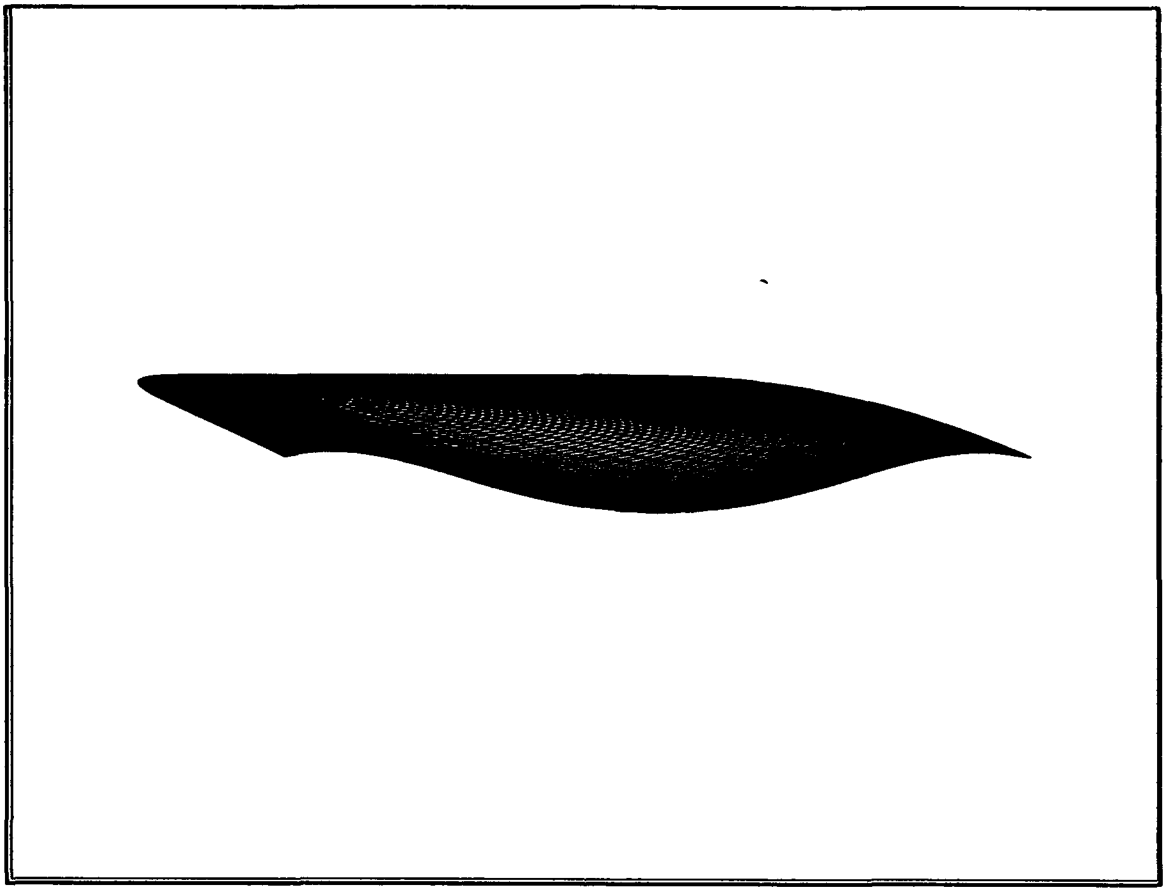

**Figure 2-4**, Waverider Model

or if the flow were to become asymmetric, which can occur for low speed flow. No such cases will be examined in this effort.

WAVERIDER contains the parameters which govern the generation of the inviscid optimized waverider: freestream Mach number, generating cone half angle and length, maximum spanwise sweep angle, and the type of parabolic freestream surface. The design parameters used here are given in Table 2.1. The surface grid dimensions are given by the integers listed in Table 2.2. Note that these values do not necessarily correspond to the number of points used in the CFD grid; they apply only to building the waverider body database around which a CFD grid is later constructed.

**Table 2-1, WAVERIDER Parameters**

| Parameter | Value | Meaning |
|---|---|---|
| MACH | 10.0 | Freestream Mach number |
| D | 5.5° | Generating cone half angle, $\delta$ |
| L | 88.41675 | Generating cone length, meters |
| PHIL | 50.0° | Maximum spanwise sweep angle, $\phi_b$ |
| TYPE | 0 | 0 indicates parabolic top waverider |

**Table 2-2, WAVERIDER Dimensions**

| Parameter | Value | Meaning |
|---|---|---|
| MCAP | 45 | Number of points which define freestream surface at a given Y-Z crossplane |
| NCAP | 90 | Number of Y-Z planes defining waverider |
| INCR | 91 | Total number of points defining waverider surface at a given Y-Z crossplane |

### 2.1.2 Subroutine WAVEBODY

This subroutine takes the equations derived by Rasmussen (25), outlined above, calculates the waverider body shape in spherical coordinates, scales the equations, and finally transforms the spherical coordinates to cartesian coordinates. The primary inputs to this subroutine are the parameters defined in the main program, WAVERIDER. The primary outputs of this program are NCAP x INCR ordered triples ( (x,y,z) coordinates) that describe the waverider in cartesian coordinates.

### 2.1.3 Subroutine OUTDATA

The subroutine OUTDATA has three primary functions. The first, as suggested by the name, is to build the output file which describes the waverider surface. The second is to use a cubic spline and linear interpolation techniques to build a blunt nose. The third is to use a cubic spline fit to blunt the leading edges.

Primary input to OUTDATA are the coordinates generated by WAVEBODY, the absolute value of the slope at the boundary points, and the number of points to use within the cubic spline fit. The latter two items are read in when the program is executed.

**Table 2-3, OUTDAT Parameters**

| Parameter | Value | Meaning |
|---|---|---|
| DIM | 15 | Number of points used to define spline, also the number of planes added to define the waverider's nose |
| SLOPE | 4.5 | Absolute value of the slope of the spline at the boundary point |

### 2.1.4 Subroutine GEOM

This subroutine, provided by Beran (8), determines the spacing of both the Y-Z crossplanes and the distribution of points within those crossplanes. Subroutine GEOM, as suggested by the name, determines the crossplane spacing via a geometric progression.

### 2.1.5 Subroutine CUBSPL

CUBSPL, developed by de Boor (14), calculates the coefficients in Eqn (2-10). Input to CUBSPL consists of IBC1, a flag that sets the boundary condition used; $\tau_i,\ldots,\tau_j$, the Z-coordinates of the endpoints of the intervals; $g_i(\tau_i),\ldots..,g_j(\tau_j)$, the Y-coordinate of the endpoints; and $s_i(\tau_i),\ldots,s_j(\tau_j)$, the slopes at the endpoints of the intervals. For I intervals there will be I+1 of each of the latter three parameters.

## 2.2 Grid Generation

The GRIDGEN software package (30) was used to generate the three-dimensional grid around the waverider body. There are three main steps in generating a grid with GRIDGEN:

1. GRIDBLOCK: In GRIDBLOCK, the boundaries of the grid are defined, as are the computational coordinates, $\xi$, $\eta$, and $\zeta$, and their associated dimensions using a Silicon Graphics IRIS 4D workstation. For the waverider, one block was used, although GRIDGEN is capable of handling multi-block configurations.

2. GRIDGEN2D: In GRIDGEN2D, the distribution of points on each of the six faces of each block is performed, resulting in the generation of surface grids on each face of each block. This step is also performed on a Silicon Graphics IRIS 4D workstation.

3. GRIDGEN3D: GRIDGEN3D is the final step, and requires a CRAY super-computer. GRIDGEN3D takes the results of GRIDGEN2D and, using one of several solution methods, generates the volumetric, three-dimensional grid. The solution is initialized with an algebraic, trans-finite interpolation method. The volume grid, once initialized, may be further refined by running one of several elliptic solvers for a given number of iterations.

For the waverider under investigation here, grid generation was performed using the following basic procedure. In GRIDBLOCK, as mentioned above, a single block defining the grid boundaries and computational coordinates was generated. The upstream boundary of the block stands off .05 meters from the nose of the waverider (see Figure 2-7) , with the domain being an elliptical cone 5 m in diameter at the nose and 25 m in diameter at the waverider's baseplane, in the Y direction. In the Z direction the elliptical cone was 5 m in radius at the nose and 12 m in radius at the baseplane. Figure 2-5 and Figure 2-9 illustrate the computational domain at the baseplane and inflow boundaries respectively. The dimensions were chosen to make the computational domain as small as possible without having the imposition of the freestream boundary conditions interfere with the flow solution. The $\xi$ coordinate points downstream from the nose, $\eta$ is normal to the waverider body, and $\zeta$ forms a right handed system, running spanwise. Grid dimensions were set to 61 x 91 x 101 in the $\xi$, $\eta$, and $\zeta$ directions, respectively.

In GRIDGEN2D, grids were generated on each face of the block created in GRIDBLOCK. The most interesting was face 5, the $\eta_{min}$ face. Face 5 was first split into two subfaces, one containing the 6 points between the nose of the waverider and the upstream block boundary, and the other being the waverider surface itself. The waverider surface grid was generated using the *lines on database networks* selection from the GRIDGEN2D subface shape menu. In other words, the surface grid was forced to conform to the waverider surface. Choice 10 from the subface interior point initialization (algebraic solver) menu, *interp u,v and fit to parametric surface*, was used to generate the surface grid.

Face 4, the $\xi_{max}$ face or waverider baseplane face, required some special attention as well. One break point was set at both the beginning and end points of the cubic spline on the waverider surface. This put 49 grid points on the compression surface, 12 were specified on the cubic spline, leaving 40 on the freestream surface. Points were clustered to the leading edge at the waverider surface, and away from the centerline. On the outer boundary a breakpoint was set almost directly above, though a little out, from the leading edge. Again, 49 points were specified for the compression region. Points on the outer boundary were clustered away from the breakpoint, and from the centerline, leading to a clustering near the middle of the two subedges. This spacing accomplished two things. First, it kept gridlines from overlapping each other, or cutting through the waverider body, and secondly, it provided for grid clustering in the region where the shock is expected to form. Figure 2-5 and Figure 2-6, on the following pages, illustrate the results of this process. Figure 2-6 illustrates the grid geometry close to the surface of the waverider.

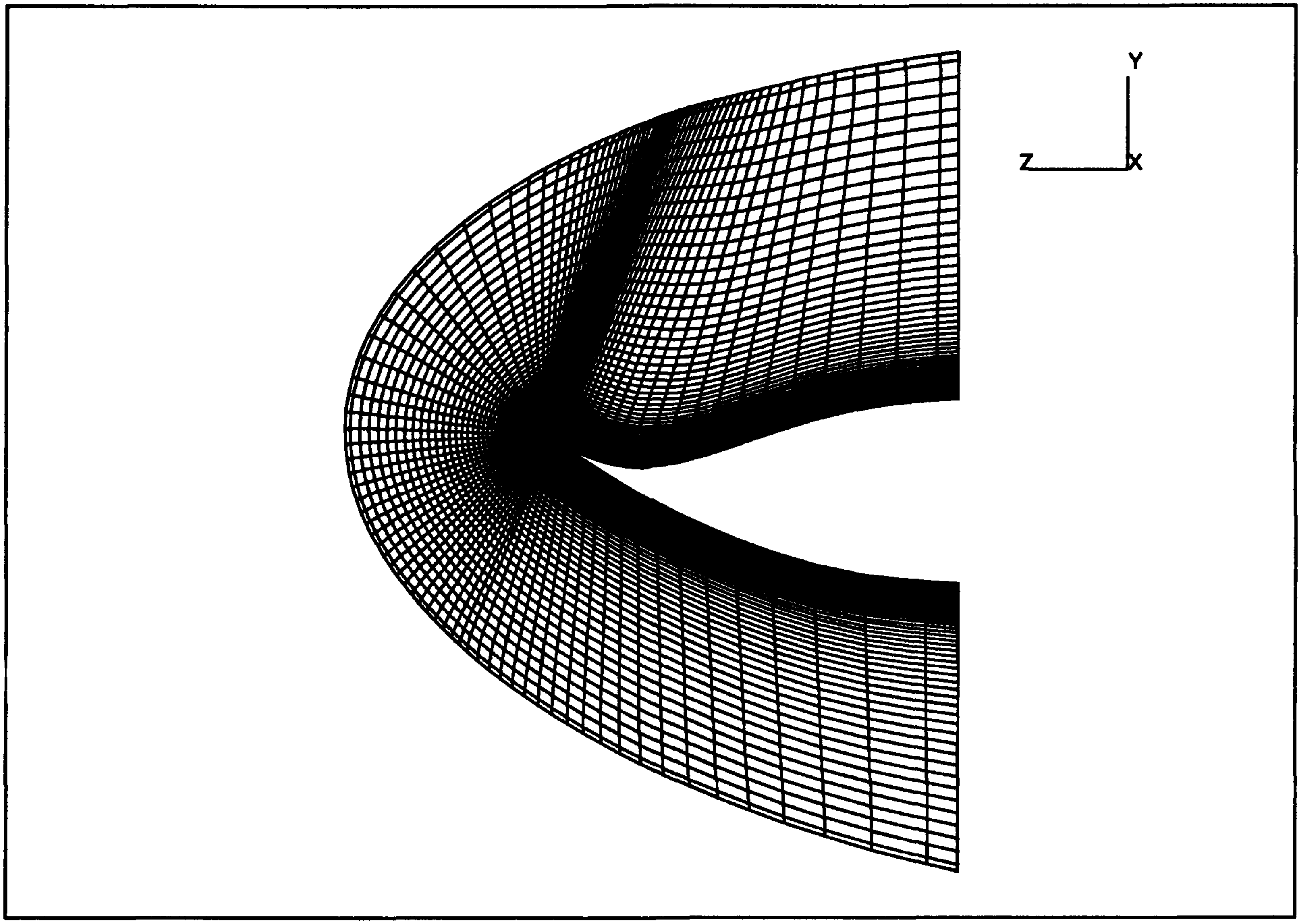

Figure 2-5, Baseplane Grid

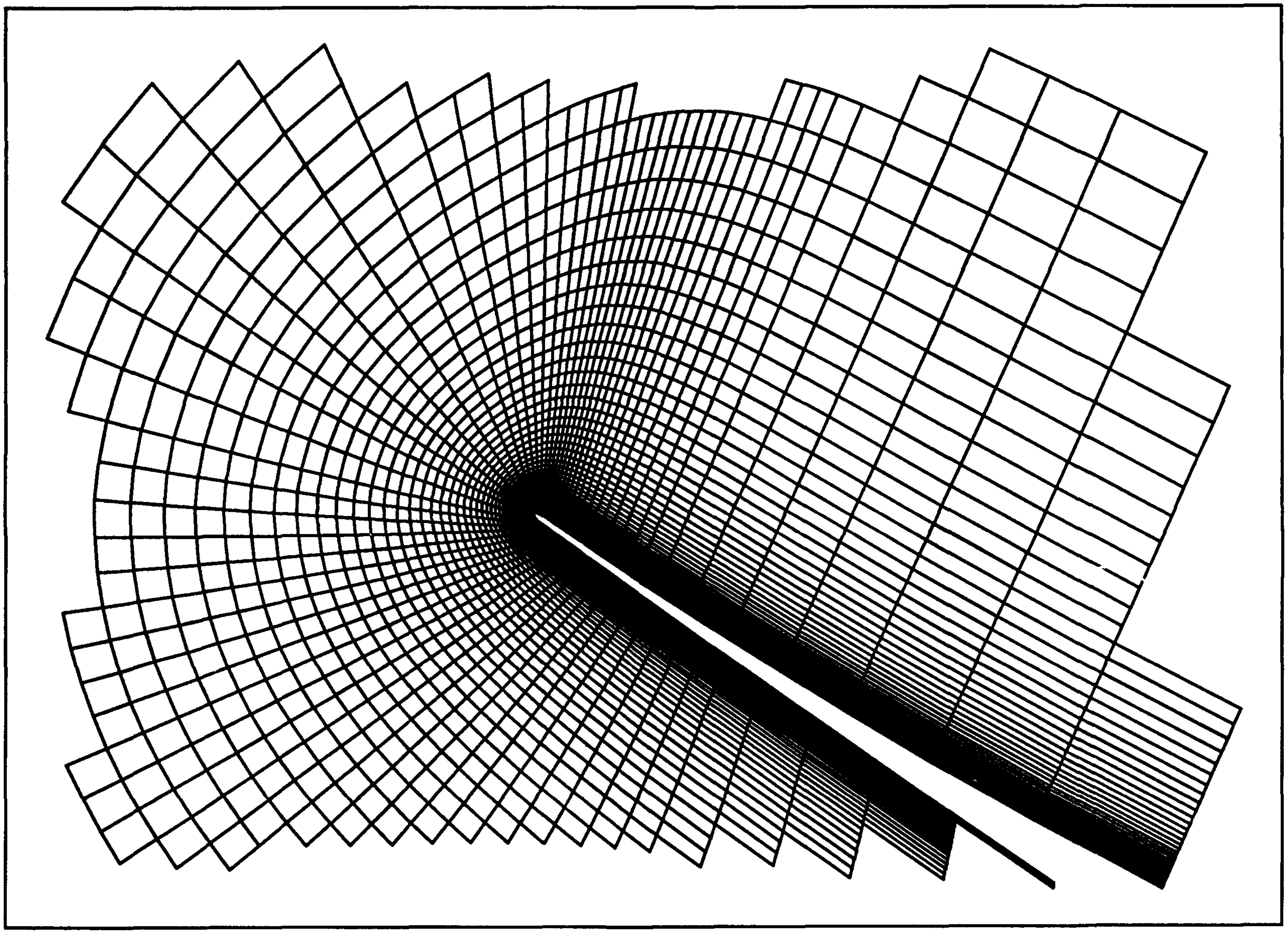

**Figure 2-6**, Baseplane Grid, Zoom of Cusp

Similar measures were taken at the nose, or freestream plane, which is face 3, the $\xi_{min}$ face. Choice 1 from the subface interior point initialization (algebraic solver) menu, *arclength based (Soni) TFI* was used to initialize this face, and all other faces except face 5.

On faces 1 and 2, the $\zeta_{min}$ and $\zeta_{max}$ faces, GRIDGEN2D's elliptic solver was used to reduce grid skewness near the waverider's nose. Thomas-Middlecoff control functions, with a relaxation factor of .25, were used for the smoothing. Thomas-Middlecoff control functions were used as they provided the best results. The small relaxation factor was chosen so that the changes to the grid would be slow enough that when the grid was acceptably smooth the process could be stopped. Approximately 200 iterations were required to produce an acceptable grid.

The rest of the faces of the block were rather straightforward, requiring little special treatment. Points were clustered toward the leading edges, the nose, and the waverider surface to improve flow resolution, particularly in the boundary layer region. Normal to the surface, the first point is spaced .5 millimeter above the surface. This spacing was arrived at by looking at a converged solution from a much coarser grid. The coarse grid's first point was .5 centimeters above the surface. At the baseplane there were about three points in the boundary layer, so the spacing in the fine grid was made one tenth of that, to provide approximately 20 points in the boundary layer at the baseplane. The following figures illustrate the grids produced by GRIDGEN2D on the various faces of the block.

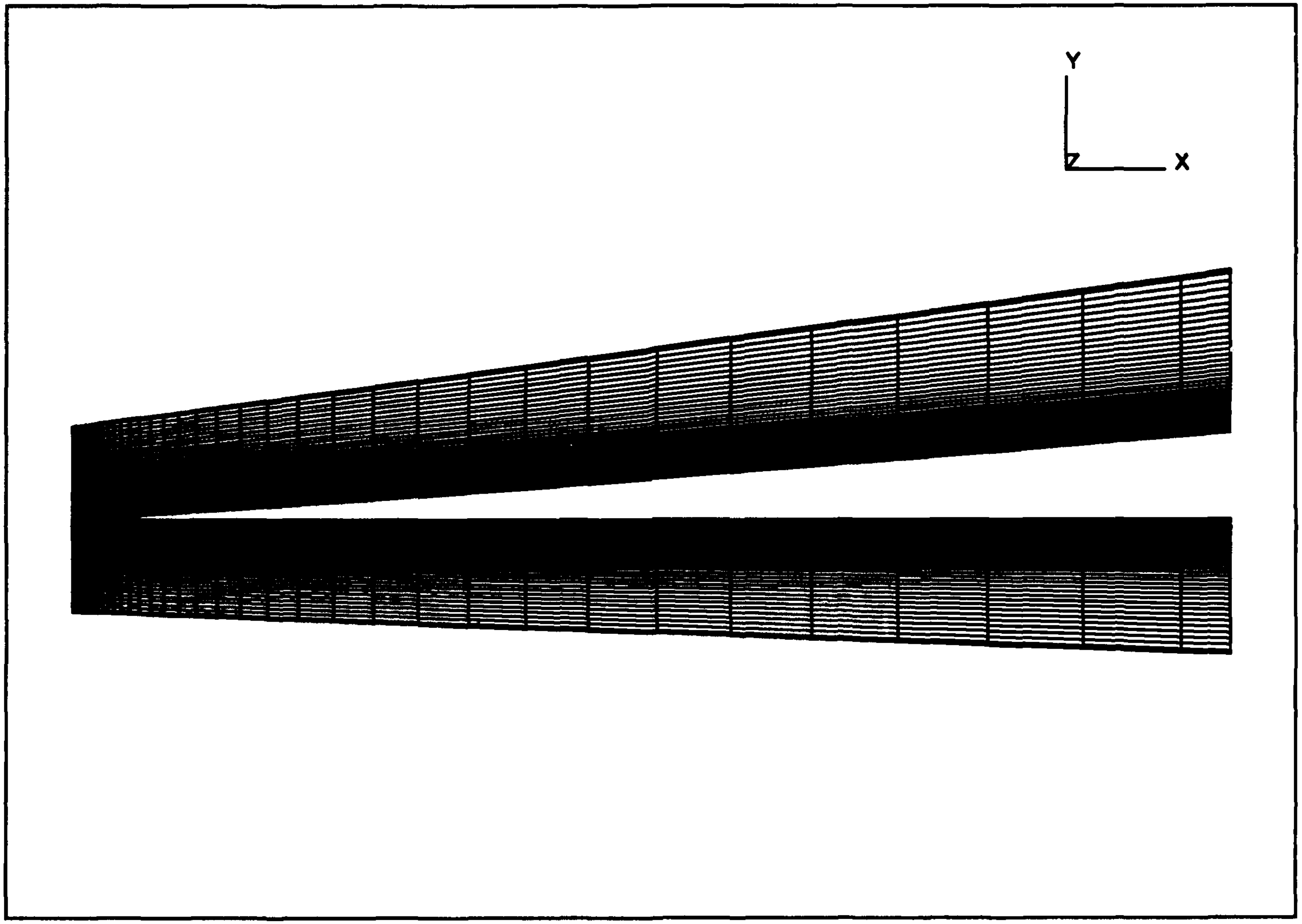

**Figure 2-7**, Centerline Grid, Faces 1 & 2

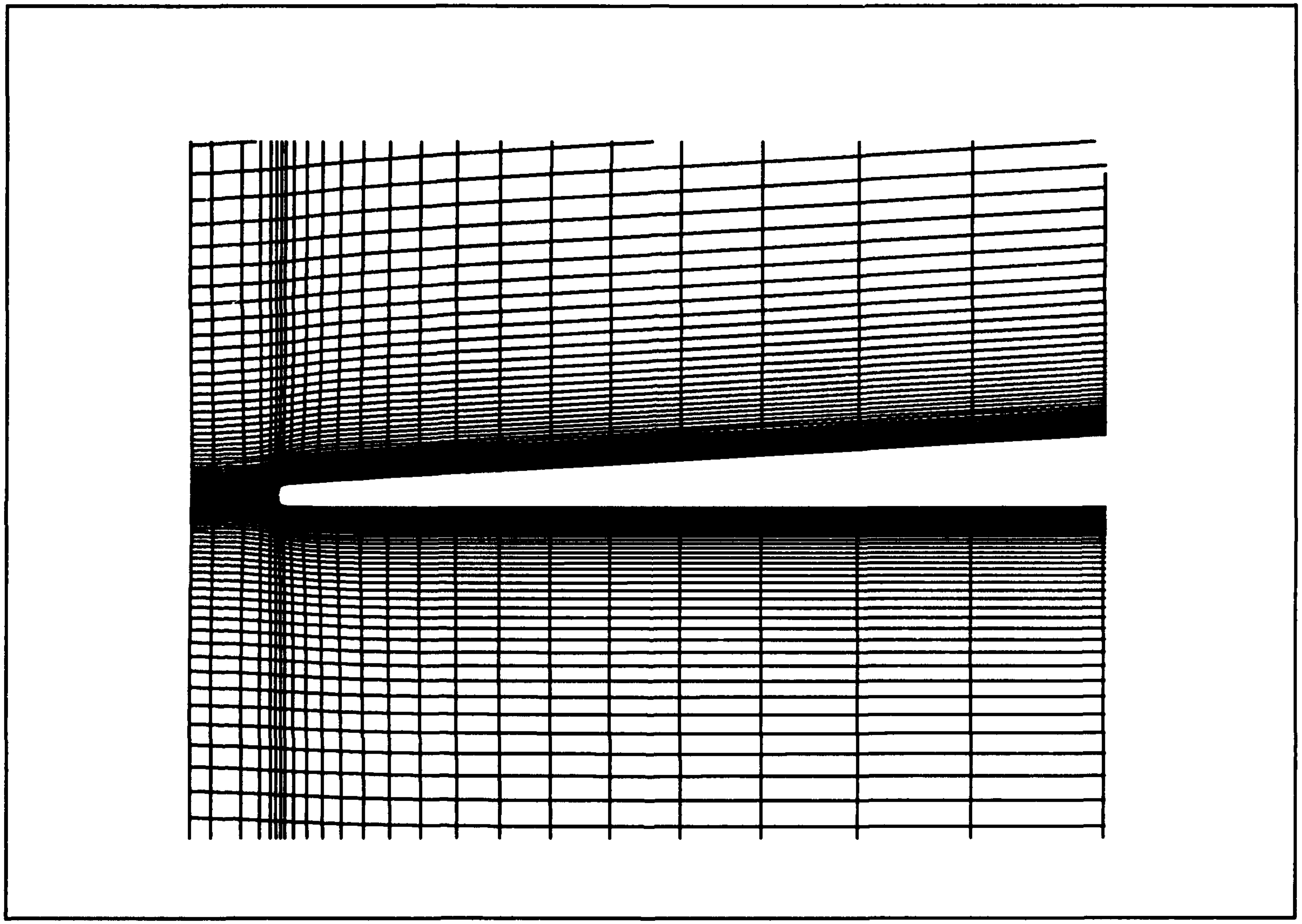

**Figure 2-8**, Zoom of Centerline Grid in Region of Nose

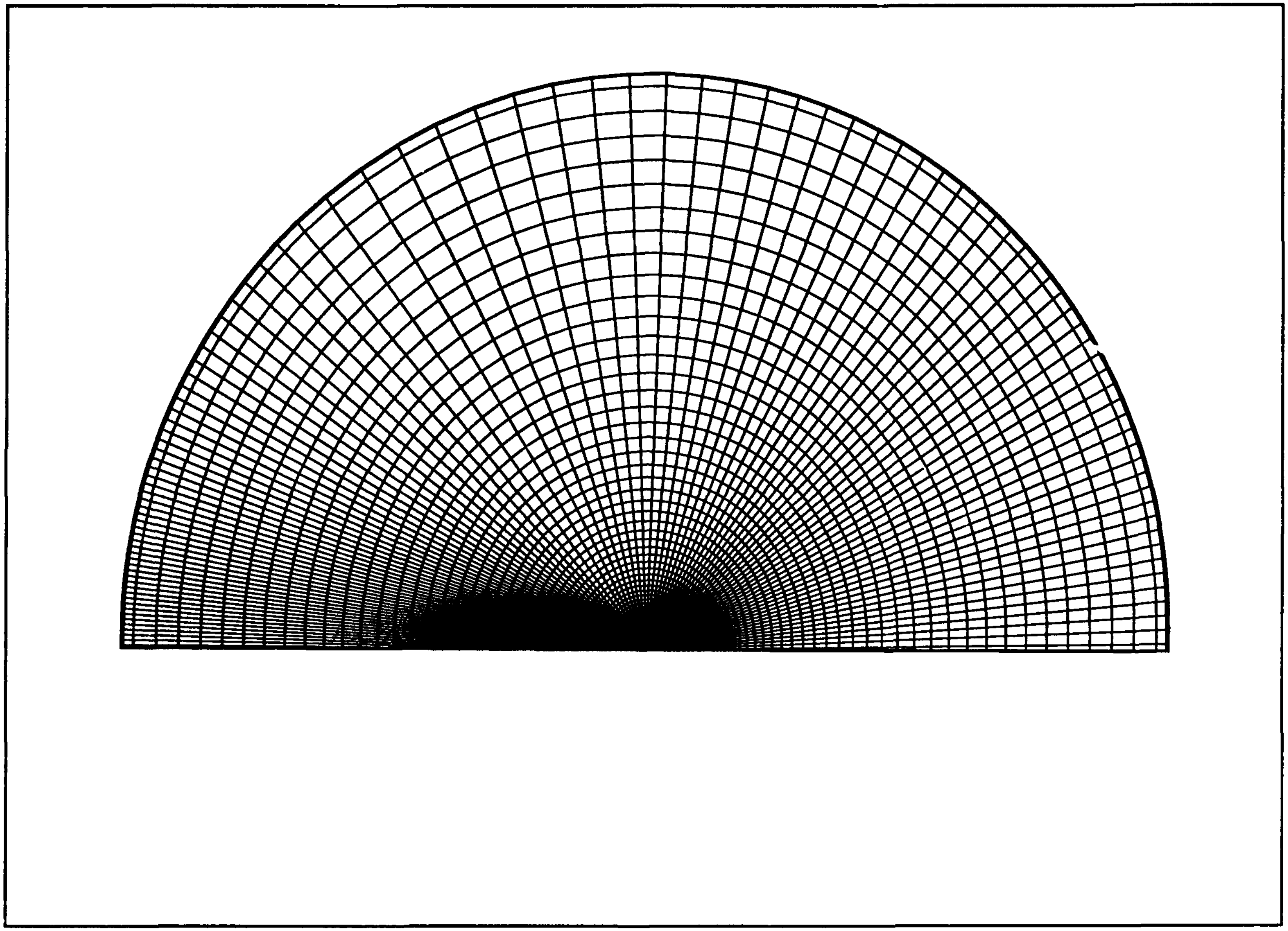

**Figure 2-9**, Freestream Grid, Face 3

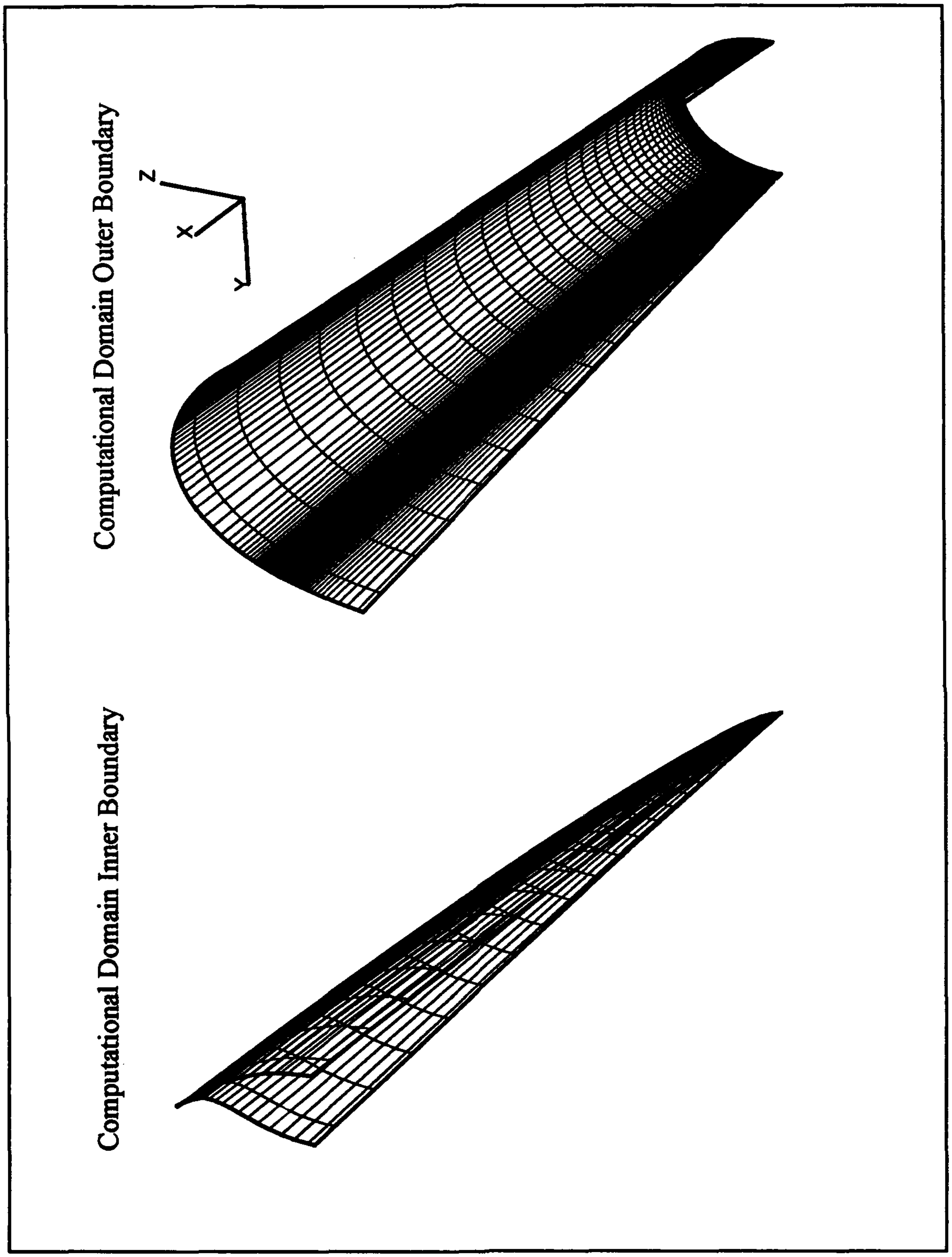
Computational Domain Outer Boundary
Z
X
Y
Computational Domain Inner Boundary

**Figure 2-10,** Inner and Outer Surface Grids, Faces 5 & 6

In GRIDGEN3D, as mentioned previously, the volume grid was initialized using an algebraic trans-finite interpolation method. This produced 385 skewed volumes and 19 negative volumes. GRIDGEN3D's elliptic solver, using Laplace control functions, was then run for 12 iterations to smooth the grid. The negative volumes were eliminated, and the number of skewed volumes remained the same. GRIDGEN3D required approximately 5 minutes to run on the CRAY Y-MP from the time the batch job was submitted until it was completed. The figures depicting the grid at the various boundaries shown on the previous pages are the output of GRIDGEN3D, with ghost points (points within the body surface) required by the computational solver added.

The ghost points mentioned above are required because finite volume codes, such as the one used here, solve for flux across cell faces, which requires knowing the flow at cell centers, rather than at the grid points. Ghost points are added to all boundaries, so that cell centers can be calculated. Figure 2-11 illustrates the addition of ghost points to a simple grid, and the resulting network of cell centers.

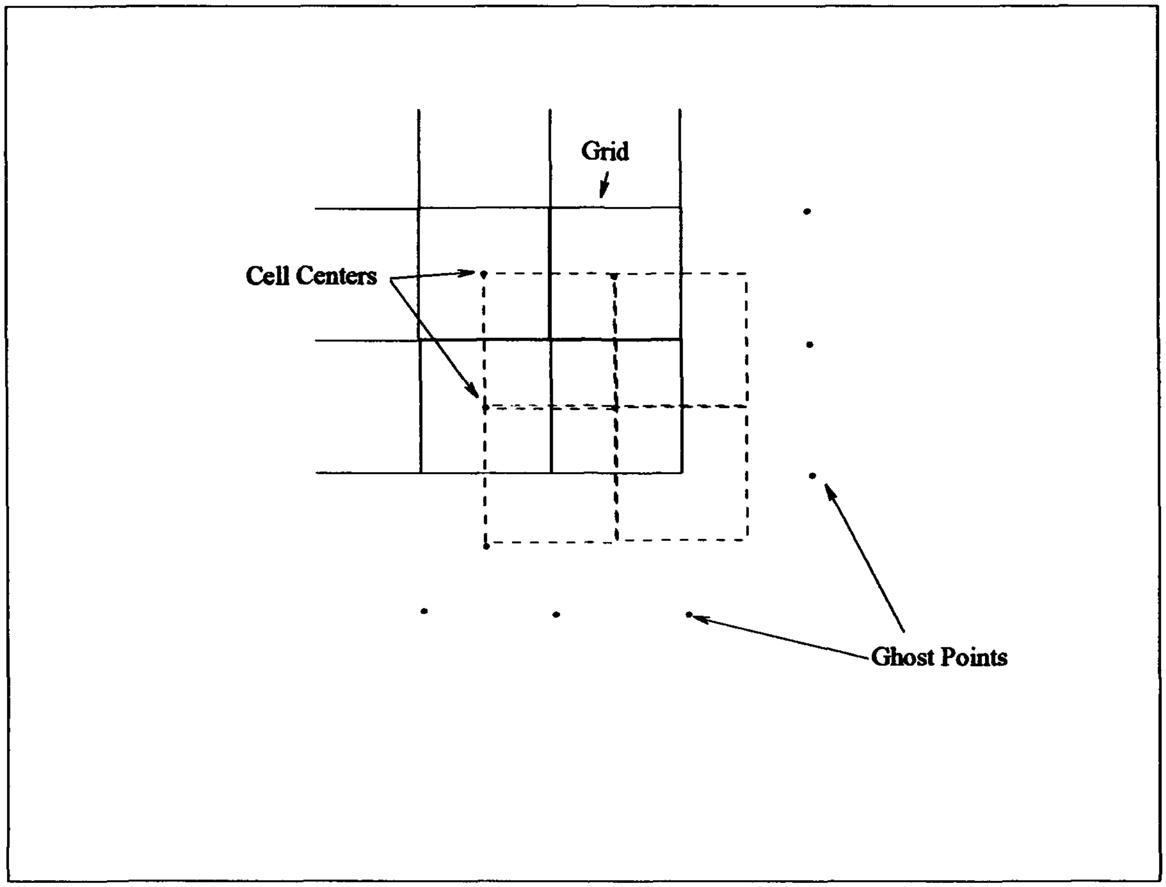

**Figure 2-11,** Addition of Ghost Points to Grid

# III. NAVIER-STOKES IMPLICIT FLUX SPLITTING ALGORITHM

## 3.1 Governing Equations

The Navier-Stokes equations are well known, and many sources provide derivations of them. Anderson (3), among others, lists the Navier-Stokes equations as

*Continuity*
$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho w}{\partial z} = 0 \qquad \textbf{(3-1)}$$

*X-Momentum*
$$\frac{D\rho u}{Dt} = -\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} \qquad \textbf{(3-2)}$$

*Y-Momentum*
$$\frac{D\rho v}{Dt} = -\frac{\partial p}{\partial y} + \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \qquad \textbf{(3-3)}$$

*Z-Momentum*
$$\frac{D\rho w}{Dt} = -\frac{\partial p}{\partial z} + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \qquad \textbf{(3-4)}$$

*Energy*
$$\begin{aligned} \frac{D\rho E_t}{Dt} &= \rho q + \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right) \\ &+ \frac{\partial(u\tau_{xx})}{\partial x} + \frac{\partial(u\tau_{xy})}{\partial y} + \frac{\partial(u\tau_{xz})}{\partial z} + \frac{\partial(v\tau_{xy})}{\partial x} + \frac{\partial(v\tau_{yy})}{\partial y} \qquad \textbf{(3-5)} \\ &+ \frac{\partial(v\tau_{zy})}{\partial z} + \frac{\partial(w\tau_{xz})}{\partial x} + \frac{\partial(w\tau_{yz})}{\partial y} + \frac{\partial(w\tau_{zz})}{\partial z} \end{aligned}$$

where $E_t$ is total energy.

The shear terms are defined as:

$$\tau_{ij} = \delta_{ij}\lambda(\nabla \cdot V) + \mu\left(\frac{\partial u_j}{\partial x_j} + \frac{\partial u_i}{\partial x_j}\right) \tag{3-6}$$

where $\delta$ is the Kronecker delta, and the subscripts correspond to standard indicial notation. Note that the shear stress terms are symmetric, i.e. $\tau_{xy} = \tau_{yx}$.

Anderson, et al. (1) presents equations (3-1) through (3-5) in vector form as

$$\frac{\partial U}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = 0 \tag{3-7}$$

where

$$U = \begin{bmatrix} \rho \\ \rho \dot{u} \\ \rho v \\ \rho w \\ E_t \end{bmatrix} \tag{3-8}$$

$$E = \begin{bmatrix} \rho u \\ \rho u^2 + p - \tau_{xx} \\ \rho uv - \tau_{xy} \\ \rho uw - \tau_{xz} \\ (E_t + p)u - u\tau_{xx} - v\tau_{xy} - w\tau_{xz} + q_x \end{bmatrix} \tag{3-9}$$

$$F = \begin{bmatrix} \rho v \\ \rho uv + p - \tau_{xy} \\ \rho v^2 - \tau_{yy} \\ \rho vw - \tau_{yz} \\ (E_t + p)v - u\tau_{xy} - v\tau_{yy} - w\tau_{yz} + q_y \end{bmatrix} \qquad \textbf{(3-10)}$$

$$G = \begin{bmatrix} \rho w \\ \rho uw + p - \tau_{xz} \\ \rho vw - \tau_{yz} \\ \rho w^2 - \tau_{zz} \\ (E_t + p)w - u\tau_{xz} - v\tau_{yz} - w\tau_{zz} + q_z \end{bmatrix} \qquad \textbf{(3-11)}$$

The form of the Navier-Stokes equations given in equations (3-7) to (3-11) is much more convenient than that in equations (3-1) through (3-5) for purposes of coding the Navier-Stokes equations into a CFD algorithm. The form of equation (3-7) is easier to code than that in equations (3-1) through (3-5) because it is more concise.

Transforming the Navier-Stokes equations to the computational coordinate system is necessary so that the equations are consistent with the computational space. The general transformed form of the Navier-Stokes equations is (1):

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{U}{J}\right) &+ \frac{\partial}{\partial \xi}\left\{\frac{1}{J}\left[\xi_x(E_i - E_v) + \xi_y(F_i - F_v) + \xi_z(G_i - G_v)\right]\right\} \\ &+ \frac{\partial}{\partial \eta}\left\{\frac{1}{J}\left[\eta_x(E_i - E_v) + \eta_y(F_i - F_v) + \eta_z(G_i - G_v)\right]\right\} \\ &+ \frac{\partial}{\partial \zeta}\left\{\frac{1}{J}\left[\zeta_x(E_i - E_v) + \zeta_y(F_i - F_v) + \zeta_z(G_i - G_v)\right]\right\} = 0\end{aligned} \qquad \textbf{(3-12)}$$

The Jacobian, ***J***, is the matrix of inverse metrics:

$$J = \frac{\partial(\xi, \eta, \zeta)}{\partial(x, y, ,z)} \qquad \textbf{(3-13)}$$

See Beran (8) for the complete statement of the Jacobian matrix.

Note in equation (3-12) that the flux vectors ***E, F*** and ***G*** have been separated into inviscid and viscous terms. The flux vector ***E***, for instance, is separated into $\boldsymbol{E_i}$ and $\boldsymbol{E_v}$, where:

$$E_i = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ (E_t + p)u \end{bmatrix} \qquad E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x \end{bmatrix} \qquad \textbf{(3-14)}$$

The viscous terms in ***F*** and ***G*** are separated from the inviscid terms in a similar fashion.

## 3.2 Discretization

The code developed at WL/FIMM and applied in the present research uses several methods in conjunction with each other to discretize the Navier-Stokes equations.

The viscous terms in the flux vectors are handled by simple centered differences. The inviscid terms are handled by a combination of flux vector splitting and flux difference splitting as described below.

Consider the one-dimensional, inviscid model equation:

$$\frac{\partial U}{\partial t} + \frac{\partial E}{\partial x} = 0 \tag{3-15}$$

First order implicit discretization of equation (3-15) with the backward Euler method produces

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} + \frac{E_{i+\frac{1}{2}}^{n+1} - E_{i-\frac{1}{2}}^{n+1}}{\Delta x} = 0 \tag{3-16}$$

where the subscript *i+1/2* denotes the interface between nodes *i* and *i+1*. Linearization of a typical flux term yields

$$\begin{aligned} E_{i+\frac{1}{2}}^{n+1} &= E_{i+\frac{1}{2}}^{n} + \left(\frac{\partial E}{\partial t}\right)_{i+\frac{1}{2}} \Delta t \\ &= E_{i+\frac{1}{2}}^{n} + \left(\frac{\partial E}{\partial U}\frac{\partial U}{\partial t}\right)_{i+\frac{1}{2}} \Delta t \\ &= E_{i+\frac{1}{2}}^{n} + \left(A^n \delta U^n\right)_{i+\frac{1}{2}} \end{aligned} \tag{3-17}$$

where

$$A \equiv \frac{\partial E}{\partial U} \qquad \textbf{(3-18)}$$

$$\delta U^{n} \equiv U^{n+1} - U^{n}$$

Substituting equation (3-17) into equation (3-16) yields:

$$\frac{\left(\delta U^{n}\right)_{i}}{\Delta t} + \frac{\left(A^{n}\delta U^{n}\right)_{i+\frac{1}{2}} - \left(A^{n}\delta U^{n}\right)_{i-\frac{1}{2}}}{\Delta x} = - \frac{E^{n}_{i+\frac{1}{2}} - E^{n}_{i-\frac{1}{2}}}{\Delta x} \qquad \textbf{(3-19)}$$

Flux vector splitting, in the form of the Steger-Warming algorithm (29), is used on the left hand side of equation (3-19), while Roe flux difference splitting (8), in the form of the MUSCL scheme (37), is used on the right hand side of equation (3-19).

The three-dimensional system in equation (3-7) is discretized very much like the one-dimensional model system above. In order to compute the right hand side, or residual, of the three-dimensional version of equation (3-19), the fluxes in each direction are successively balanced. Thus

$$LHS_{1} = \frac{E_{i+\frac{1}{2}}(U^{n}) - E_{i-\frac{1}{2}}(U^{n})}{\Delta x}$$

$$LHS_{2} = LHS_{1} + \left(\frac{F_{i+\frac{1}{2}}(U^{n}) - F_{i-\frac{1}{2}}(U^{n})}{\Delta y}\right) \qquad \textbf{(3-20)}$$

$$LHS_{3} = LHS_{2} + \left(\frac{G_{i+\frac{1}{2}}(U^{n}) - G_{1-\frac{1}{2}}(U^{n})}{\Delta z}\right)$$

where LHS$_1$ refers to the left hand side of the three-dimensional version of equation (3-19).

The left-hand side of equation (3-20) is calculated using Steger-Warming flux-vector splitting (29), discussed in the following section. Roe's upwind method (8) is used on the right hand side of equation (3-20) to calculate $E_{i+1/2}$, $F_{i+1/2}$, and $G_{i+1/2}$. For example:

$$E_{i+\frac{1}{2}} = \frac{1}{2}\left[E(U_L) + E(U_R) - \frac{1}{2}\hat{A}\left(U_R - U_L\right)\right] \qquad \textbf{(3-21)}$$

The terms $U_R$ and $U_L$ are obtained from the MUSCL approach described in Section 3.4. The "^" indicates a Roe averaged term, also described in Section 3.4

## 3.3 Flux Vector Splitting

Flux vector splitting is a well known technique used to improve the computational efficiency of finite-difference schemes, as well as to make the schemes somewhat more robust. The basic aim is to split the flux vectors such that an upwind finite-difference scheme may be used at all points within the flow. This is done by the simple expedient of separating the flux vectors into upwind and downwind parts. In WL/FIMM's high speed, implicit, flux-difference splitting CFD code is used on the left-hand side of the finite difference equation, i.e. the left hand side of equation (3-20). Steger & Warming (29) introduce flux vector splitting in some detail. Briefly, for the three-dimensional Navier-Stokes equations, flux-vector splitting proceeds as follows.

Equation (3-7), neglecting viscous terms, can be rewritten as

$$\frac{\partial U}{\partial t} + A\frac{\partial U}{\partial x} + B\frac{\partial U}{\partial y} + C\frac{\partial U}{\partial z} = 0 \qquad \textbf{(3-22)}$$

where ***A*, *B*,** and ***C*** are the Jacobian matrices

$$A = \frac{\partial E_i}{\partial U} \qquad B = \frac{\partial F_i}{\partial U} \qquad C = \frac{\partial G_i}{\partial U} \tag{3-23}$$

which correspond to the inviscid portions of the flux vectors, $E_i$, $F_i$, and $G_i$. Note that $E_i=AU$, $F_i=BU$, and $G_i=CU$ by the first order homogenous property of the Euler equations.

A finite-difference form of equation (3-22) is:

$$\frac{\delta U_i^n}{\Delta t} + \frac{D\left(A_i^n \delta U_i^n\right)}{\Delta x} + \frac{D\left(B_i^n \delta U_i^n\right)}{\Delta y} + \frac{D\left(C_i^n \delta U_i^n\right)}{\Delta z} = -\left[\frac{DE_i^n}{\Delta x} + \frac{DF_i^n}{\Delta y} + \frac{DG_i^n}{\Delta z}\right] \tag{3-24}$$

where δ and ***D*** are the time and space difference operators, respectively:

$$\delta f \equiv f^{n+1} - f^n \tag{3-25}$$

$$Df \equiv f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}} \tag{3-26}$$

Now, define the matrix ***P*** as a linear combination of ***A*,*B*,** and ***C***:

$$P = k_1 A + k_2 B + k_3 C \tag{3-27}$$

where the $k_i$ are arbitrary real constants.

The system in equation (3-22) is hyperbolic if there exists a similarity transformation such that

$$Q^{-1}PQ = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & 0 \\ 0 & 0 & 0 & 0 & \lambda_5 \end{bmatrix} = \Lambda \qquad \textbf{(3-28)}$$

The (general) flux vector is given by

$$\mathcal{F} = Q\Lambda Q^{-1}U \qquad \textbf{(3-29)}$$

Note that the Jacobian can be diagonalized due to the hyperbolic nature of the inviscid fluxes. The eigenvalues, $\lambda_i$, of the 5x5 matrix, $\Lambda$, are arbitrary, but real. The matrix $\boldsymbol{Q}$ and its inverse are calculated as follows:

$$Q = MT \qquad Q^{-1} = T^{-1}M^{-1} \qquad \textbf{(3-30)}$$

The matrices $\boldsymbol{M}$, $\boldsymbol{T}$, and their inverses are given by Warming (35).

The splitting comes in through the eigenvalue matrix, $\Lambda$. It is split into two matrices, $\Lambda^+$ and $\Lambda^-$ via the splitting of the eigenvalues, $\lambda_i$:

$$\lambda_i^+ = \tfrac{1}{2}(\lambda_i + |\lambda_i|) \qquad \lambda_i^- = \tfrac{1}{2}(\lambda_i - |\lambda_i|) \qquad \textbf{(3-31)}$$

Thus, the flux vector is split by using $\Lambda^+$ and $\Lambda^-$ in place of $\Lambda$ in equation (3-29), so:

$$\mathcal{F} = Q\Lambda^+Q^{-1}U \qquad \mathcal{F} = Q\Lambda^-Q^{-1}U \qquad \textbf{(3-32)}$$

Hence, in equation (3-24) the terms operated on by the $D/\Delta x$ operator can be rewritten as:

$$\frac{D}{\Delta x}\left(A_i{}^n \delta U_i{}^n\right) = A^n_{i+\frac{1}{2}} \delta U^n_{i+\frac{1}{2}} - A^n_{i-\frac{1}{2}} \delta U^n_{i-\frac{1}{2}}$$

$$= A_i^+ \delta U_i + A^-_{i+1} \delta U_{i+1} - \left(A^+_{i-1} \delta U_{i-1} + A_i^- \delta U_i\right) \tag{3-33}$$

### 3.4 Roe Algorithm

Beran (8), presents the Roe scheme for a one dimensional system of equations such as in equation (3-15), as

$$U_i^{n+1} = U_i^n - \lambda\left(H^n_{i+\frac{1}{2}} - H^n_{i-\frac{1}{2}}\right)$$

$$H^n_{i+\frac{1}{2}} = \frac{1}{2}\left(E^n_{i+1} + E_i^n - \hat{R}|\hat{\Lambda}|\hat{R}^{-1}\left(U_{i+1} - U_i^n\right)\right) \tag{3-34}$$

As mentioned in Section 3.2, the code developed by WL/FIMM uses a variation of the Roe scheme, Monotonic Upstream Scheme for Conservation Laws, or MUSCL. MUSCL is a form of a TVD, total variation decreasing, scheme. Yee (37) presents the MUSCL scheme using a simple, one dimensional hyperbolic equation. Starting from equation (3-34) the MUSCL scheme replaces the $\boldsymbol{U_{i+1}}$ and $\boldsymbol{U_i}$ terms with $\boldsymbol{U^R_{i+1/2}}$ and $\boldsymbol{U^L_{i+1/2}}$ respectively. $\boldsymbol{U^R}$ and $\boldsymbol{U^L}$ are given by

$$U^R_{i+\frac{1}{2}} = U_{i+1} - \frac{1}{4}\left[\left(1 - \bar{\eta}\right)\tilde{\Delta}_{i+\frac{3}{2}} + \left(1 + \bar{\eta}\right)\Delta_{i+\frac{1}{2}}\right] \tag{3-35}$$

and

$$U^{L}_{i+\frac{1}{2}} = U_i - \frac{1}{4}\left[(1 - \bar{\eta})\Delta_{i-\frac{1}{2}} + (1 + \bar{\eta})\tilde{\Delta}_{i+\frac{1}{2}}\right] \tag{3-36}$$

where:

$$\bar{\eta} = \begin{cases} -1: & \textit{upwind scheme} \\ 0: & \textit{Fromm scheme} \\ 1/3: & \textit{third order upwind biased scheme} \\ 1: & \textit{three point central difference scheme} \end{cases} \tag{3-37}$$

$$\tilde{\Delta}_{i+\frac{1}{2}}U = \mathit{minmod}\left(U_{i+1} - U_i, \omega\left(U_i - U_{i-1}\right)\right) \tag{3-38}$$

$$\Delta_{i+\frac{1}{2}}U = \mathit{minmod}\left(U_{i+1} - U_i, \omega\left(U_{i+2} - U_{i+1}\right)\right) \tag{3-39}$$

$$1 \le \omega \le \frac{3 - \bar{\eta}}{1 - \bar{\eta}} \qquad \bar{\eta} \ne 1 \tag{3-40}$$

and, finally

$$\mathit{minmod}(x,y) = \mathit{sgn}(x)\cdot\max\left\{0, \min\left[|x|, y\,\mathit{sgn}(x)\right]\right\} \tag{3-41}$$

Note that *sgn(x)* means the sign of the variable *x*. Note also that the *minmod* slope limiter, equation (3-41), is not the only limiter that may be applied.

The "^" in the above equations refers to Roe averaging, which is given by:

$$\hat{u} = \frac{\sqrt{\rho_L}\; u_L + \sqrt{\rho_R}\; u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{3-42}$$

The *L* and *R* subscripted variables in equation (3-42) refer to the components of the like subscripted vector $\boldsymbol{U}$ in equations (3-35) and (3-36).

### 3.5 Boundary and Initial Conditions

There are four basic boundary conditions imposed in the waverider problem: freestream conditions, surface conditions, symmetry conditions and no-change conditions. The initial conditions were simply uniform flow at freestream conditions.

The freestream conditions were imposed on the $\xi_{min}$ face and the $\eta_{max}$ face. Fluxes on the freestream boundaries were calculated using the values for Mach number, temperature and pressure assigned in the initial conditions. On the $\xi_{min}$ face, the inflow condition is specified by

$$U_{1,j,k} = U_\infty \qquad \textbf{(3-43)}$$

where $\boldsymbol{U}$ is the vector given in equation (3-8).

Similarly, the freestream condition is given on the $\xi_{max}$ face by

$$U_{i,j_{max},k} = U_\infty \qquad \textbf{(3-44)}$$

The boundary conditions at the waverider's surface are a bit more complicated. The viscous terms are calculated directly from the conditions specified at the surface for temperature and zero velocity. Because the computational grid is cell centered, the conditions at the surface are imposed by setting the conditions at the ghost points just within the surface. Thus, for the *u*-component of velocity

$$u_{i,1,k} = -u_{i,2,k} \tag{3-45}$$

identical relations are used for the other velocity components, $v$, and $w$. Furthermore, pressure is held constant between the ghost point and the first point above the surface:

$$P_{i,1,k} = P_{i,2,k} \tag{3-46}$$

Temperature at the ghost point is calculated using:

$$T_{i,1,k} = 2\,T_{wall} - T_{i,2,k} \tag{3-47}$$

The inviscid fluxes are calculated directly from the pressure and the metrics. Since the inviscid energy and mass fluxes normal to the surface are both zero, the fluxes are given by:

$$U_{i,wall,k} = \begin{bmatrix} 0 \\ P\eta_x \\ P\eta_y \\ P\eta_z \\ 0 \end{bmatrix} \tag{3-48}$$

Recall from above that the pressure is assumed constant between the ghost point and the first point above the surface. Note also that the metrics are calculated at the wall, since wall points are actually grid points.

The symmetry boundary condition is imposed on the $\zeta_{min}$ and $\zeta_{max}$ faces. Flux in the $\zeta$ direction is set to zero by setting the flux at the ghost points equal to the negative of the flux at the first point out from the symmetry plane. For the $\zeta_{min}$ face this is:

$$\rho w_{i,j,1} = -\rho w_{i,j,2} \tag{3-49}$$

with the exact same form for the $\zeta_{max}$ face.

A no-change condition is applied to the outflow boundary, the $\xi_{max}$ face. The no-change condition is imposed by simply setting the fluxes at the ghost point equal to those at the first point upstream of the boundary, thus:

$$U_{i_{max}+1,j,k} = U_{i_{max},j,k} \tag{3-50}$$

When calculating quantities such as the lift to drag ratio, L/D, the calculations were simply terminated at the outflow boundary.

### 3.6 Computer Code Description

Wright Laboratory's three-dimensional, implicit, flux-splitting, Navier-Stokes, Fortran code developed by Gaitonde (15) under contract to WL/FIMM, was used to solve the viscous, hypersonic flow over a Mach 10, inviscid-optimized waverider, modified for viscous flow. The code was debugged on the ASD CRAY X-MP supercomputer using a Silicon Graphics IRIS workstation as an interface. The debugged code was transferred to an IRIS workstation in the AFIT computer laboratory from which it was sent to the CRAY Y-MP at the Ohio Supercomputer Center (OSC), where all solution runs were performed. Access to the OSC CRAY was through a grant for research on *Numerical Solution of Inviscid/Viscous Hypersonic Flow Around a Conically Derived Waverider*, grant number PIS009-2. Connection to the OSC CRAY was made via telnet from the AFIT IRIS workstations.

Two data files were required for execution of the code: *cn1dat* and *cn1grd*. The first file, *cn1dat*, contains information such as the version of the code to run (Navier-Stokes or Euler), whether to use an implicit or an explicit formulation, initial conditions, boundary conditions, and so forth. The *cn1dat* file has essentially four parts: solution integration parameters, flow field conditions, boundary conditions, and control parameters. A listing, by category, of the input data file follows. A sample *cn1dat* file can be found in Appendix D.

The first table, Table 3-1, lists the parameters that control the implementation of the solution, particularly such things as which finite difference scheme to use, i.e., Roe, Lax-Wendroff, or Van Leer. Other parameters include the ending iteration number, whether the flow model is Euler or Navier-Stokes, what the CFL stability criteria is, and how it is handled. Table 3-2 details the flow field conditions, particularly the freestream conditions and the orientation of the waverider, along with some data about the waverider itself, such as surface temperature. Table 3-3 lists the boundary condition parameters, and the node range in which they apply.

**Table 3-1, Solution Integration Parameters**

| | |
|---|---|
| ICASE | Finite Difference Scheme: 1=Roe |
| NEND | Number of iterations |
| INS | Flow model: 1=Navier-Stokes, 0=Euler |
| IL,JL,KL | Grid Dimensions |
| CFLMAX | Maximum CFL number allowed |
| CFL | Starting CFL number |
| CFLEXP | Number of iterations before CFL doubles |
| ICFL | Number of iterations between CFL increases |
| IPC | Driver: 1=Backward Euler, 1=Predictor-Corrector |
| IMPLT | Implicit vs. Explicit: 0=Explicit, 1=Implicit |

**Table 3-2, Flow Field Conditions**

| | |
|---|---|
| IADBWL | Flag to indicate adiabatic wall BC |
| ALPHA | Angle of attack |
| PHI | Yaw angle |
| TWALL | Wall temperature |
| RMINF | Freestream Mach number |
| REL | Reynolds number per foot |
| RLSCL | Scaling factor |
| TINF | Freestream Temperature |

**Table 3-3, Boundary Conditions**

| | |
|---|---|
| IBC | Start and end points for a $\xi$ BC |
| JBC | Start and end points for an $\eta$ BC |
| KBC | Start and end points for a $\zeta$ BC |
| IBCTYPE | Boundary condition type imposed |

**Table 3-4, Control Parameters**

| | |
|---|---|
| IREAD | 0=Deadstart, 1=Restart |
| IGRID | Grid format |
| IP3DOP | Plot3D output file format |
| MODPR | Iterations between printing convergence data |

The file *cnlgrd* contains, in binary format, the grid generated using GRIDGEN (as described in Chapter 2). The GRIDGEN output was converted to a cell centered grid by the routine REDUCE.F, supplied by Gaitonde. The coordinate axes defined in GRIDGEN, $\xi$ = streamwise axis, $\eta$ = surface normal axis, and $\zeta$ = spanwise axis, were maintained.

# IV. COMPUTATIONAL RESULTS

## 4.1 Inviscid Flow Results

The design point for the inviscid optimized waverider being studied is Mach 10 at 100,000 feet, identical to that studied by Stecklein (28). Table 4.1 lists the design parameters. The design point should, for the inviscid case, produce the best lift to drag (L/D) ratio possible for the waverider as that is the condition the waverider was optimized for.

**Table 4-1, Waverider Design Parameters**

| Parameter | Value |
|---|---|
| Mach Number | 10 |
| Altitude | 100,000 ft |
| Freestream Temperature | 406.7 °R |
| Wall Temperature | 530 °R |
| Freestream Density | 3.32 E-05 slugs/ft$^3$ |
| Freestream Pressure | 22.43 lbs/ft$^2$ |

### 4.1.1 Unmodified Waverider

In order to establish consistency between the implicit version of the CFD algorithm and the explicit version, an inviscid case, using a waverider model and grid identical to that used by Stecklein (28), was run to a fully converged solution. Convergence is defined as occurring when the residual drops below $10^{-5}$. The residual is given by

For plotting purposes, the residual as defined in equation (4-1) was normalized by the

$$G.R. = \frac{1}{(IL)(JL)(KL)}\sqrt{\sum_{i=1}^{IL}\sum_{j=1}^{JL}\sum_{k=1}^{KL}\sum_{l=1}^{5}\left(\frac{(\Delta U_l)_{i,j,k}}{U_{\infty,l}}\right)^2} \tag{4-1}$$

value calculated for the first iteration.

Stecklein (28) computed a value of 8.18812 for L/D in his work, using the explicit version of the code. The implicit version of the code returned a value of 8.18813. The two values differed in the fifth decimal place.

The main difference between the results from the implicit and explicit versions of the code is the convergence history. The implicit case was fully converged, using local time stepping, within about 300 iterations, and an argument could be made that convergence was reached in only 250 iterations, as illustrated by Figure 4-1.

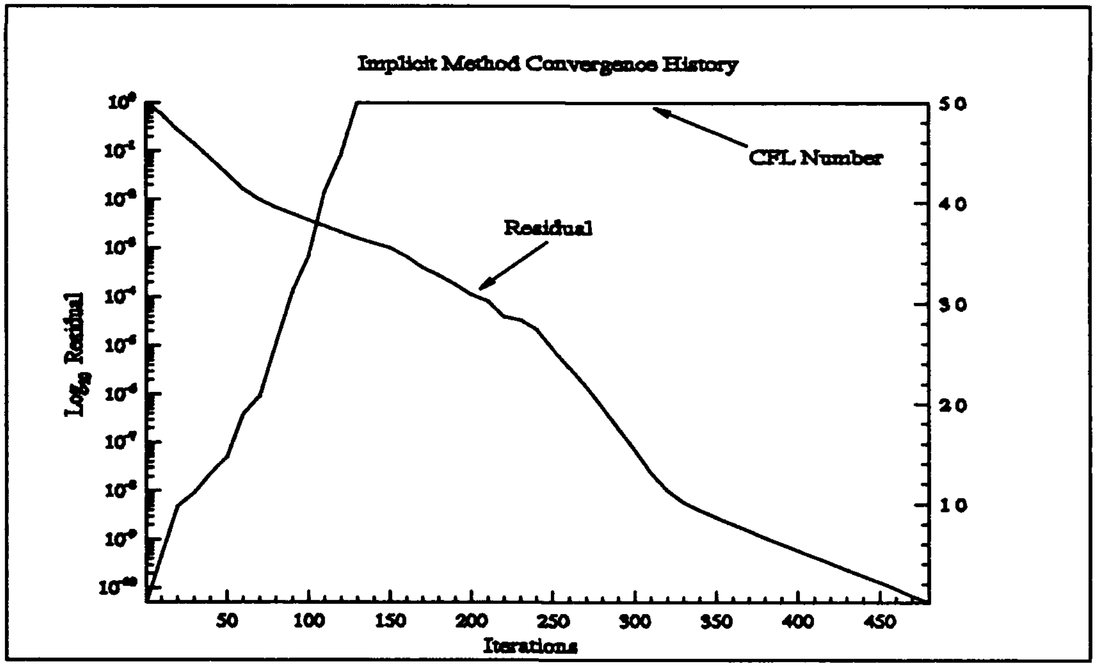

**Figure 4-1**, Inviscid Waverider Convergence History

In comparison, the explicit version of the code required over 950 iterations, with local time stepping, to reach convergence (Figure 4-2). The fewer number of iterations required by the implicit method shows its strength.

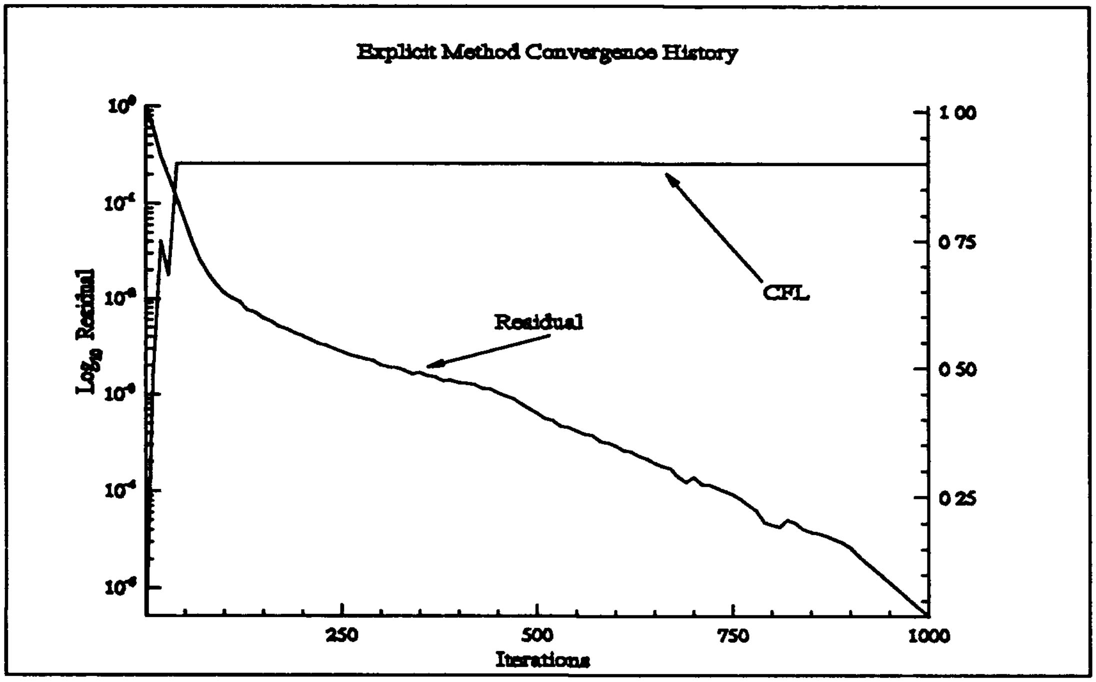

**Figure 4-2**, Inviscid Waverider Convergence History

The CFL number shown in the above figures is a constant used in determining the time step, as shown by Equations (4-2) through

$$\Delta t_{\xi} = \mathrm{CFL}\,\min_{i,j,k} \Delta t_{1} \tag{4-2}$$

where

$$\Delta t_{1} = \frac{\Delta s_{\xi}}{|u_{\xi}| + c + \dfrac{2\theta}{\Delta s_{\xi}\rho}} \tag{4-3}$$

and

$$\theta = \max\left\{ |2\mu - \lambda|, \gamma\frac{\mu}{Pr} \right\} \qquad \textbf{(4-4)}$$

The variable $s$ is the distance across the cell, with the subscript indicating direction. The variable $c$ is the local speed of sound.

The drawback to the implicit version is that each iteration of the implicit method takes considerably more computer time than each iteration of the explicit method. The implicit version of the code took 0.0001 CPU seconds per iteration per node. The explicit version required approximately 0.000054 CPU seconds per iteration per node, approximately twice as fast as the implicit version. The explicit version, however, requires only 85% of the memory that the implicit version does.

The shock capturing ability of both methods is essentially equivalent. Contour plots of density on a Y-Z plane near the base plane (Figure 4-3 and Figure 4-4) shows a very small difference between the results of the two methods. Figure 4-3 and Figure 4-4 show the results for the Y-Z plane just upstream of the baseplane. Note that the shock is, effectively, entirely captured by the compression surface of the waverider. There is a slight expansion, or spillage, at the leading edge due to numerical truncation of the waverider model, and the numerical error of the solution. The two figures are very much alike, which is comforting, showing that the two methods reached the same solution. The implicit method seemed to do a better job of capturing the shock, though that, as the different placement of the contour line closest to the waverider surface, may be due to the different convergence the two solutions went to. The results shown in Figure 4-4 are essentially identical to those shown in Figure 4-3. There is a slight difference in the placement of three of the contour lines, but that can be attributed to the different residuals of the two cases. Basically, the implicit and explicit methods produced the same results. L/D differed in the 5th decimal place, the shocks formed in the same place, and the distributions of the flow variables were practically identical.

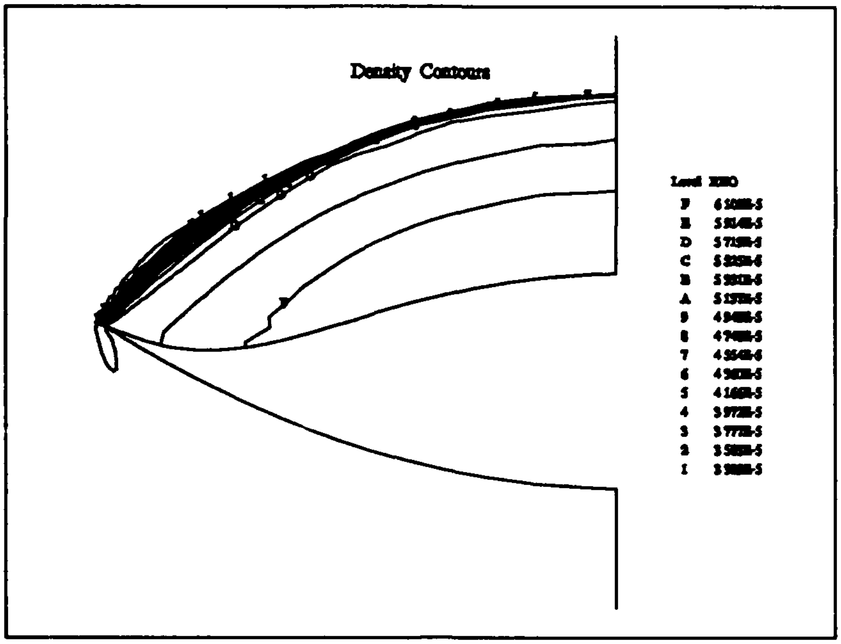

**Figure 4-3**, Implicit Method, Inviscid Waverider

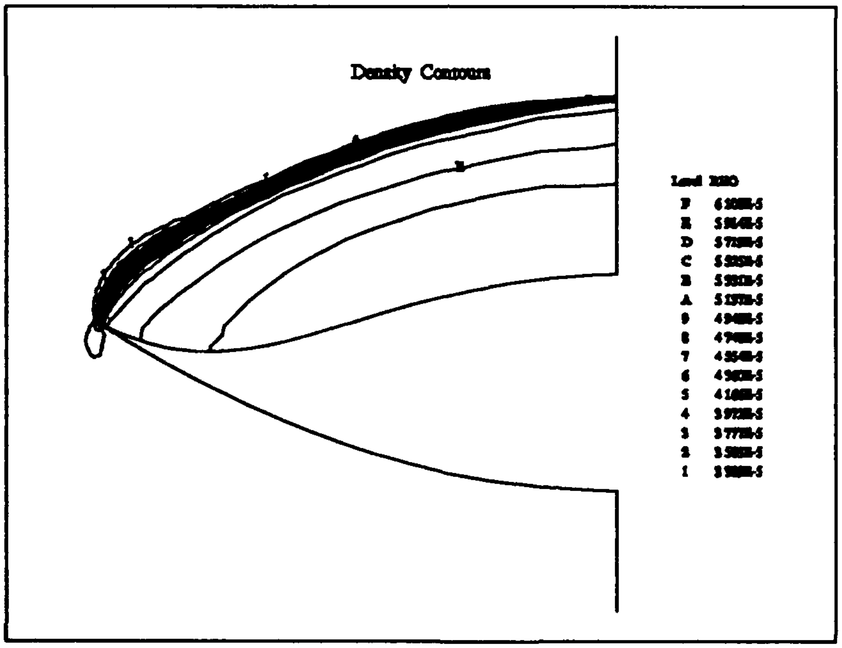

**Figure 4-4**, Explicit Method, Inviscid Waverider

### 4.1.2 Modified Waverider

An inviscid computational solution for the waverider modif'ed for viscous flow gave results very similar to the unmodified waverider. A sparser grid than that used for the Navier-Stokes case was used in an effort to conserve computer resources. The implicit method was used exclusively on the modified waverider, again, in an effort to conserve computer resources. The L/D ratio was slightly lower than for the unmodified waverider, due mainly to a rise in drag. The L/D ratio was calculated to be 8.158, only 0.4% less than that calculated by Stecklein (28) for the inviscid optimized waverider. Both lift and drag changed, drag most noticeably. The change to lift was practically unnoticeable. A contour plot of density at the base plane, Figure 4-5, shows the shock forming similarly to the unmodified waverider model, Figure 4-3. The same flow spillage seen in Figure 4-3 and Figure 4-4 was noted at the leading edges, although now it is caused more by the perturbation of the flow by the blunt leading edge than by numerical errors in the waverider model and/or the solution.

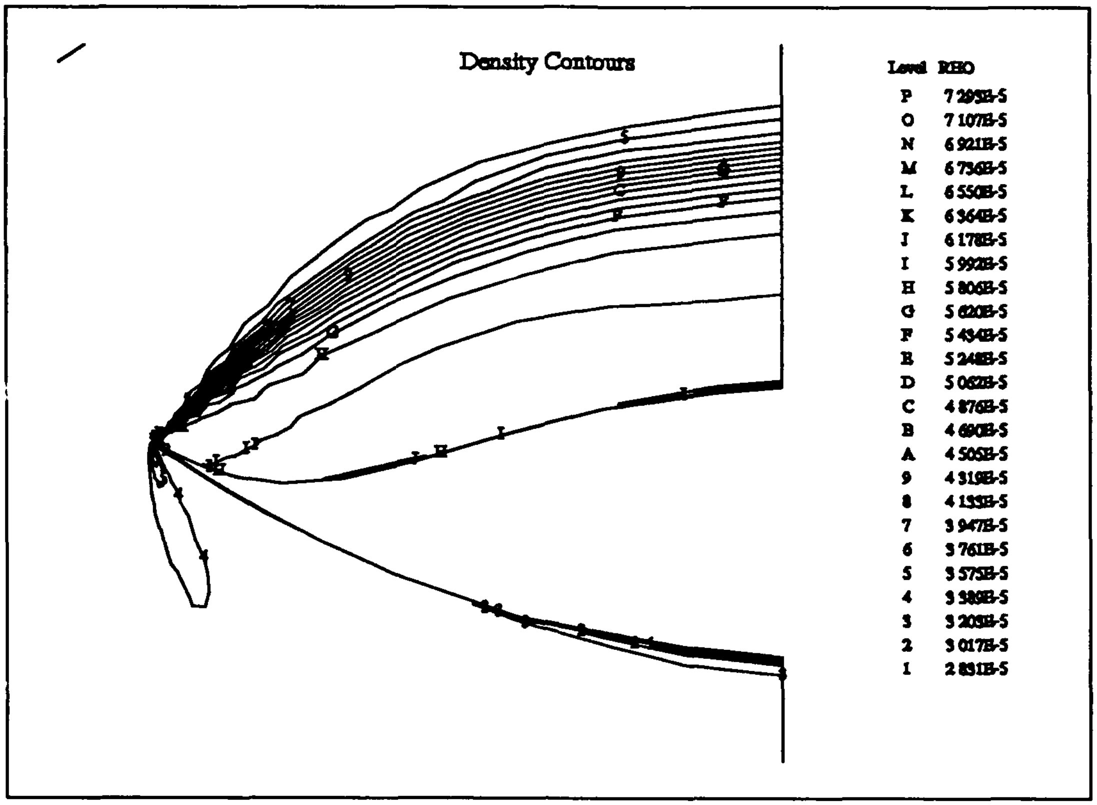

**Figure 4-5**, Modified Waverider Model, Inviscid Flow

The blunted nose produced a detached bow shock. This detached bow shock produced very high temperatures in the nose region, since the shock was, in that region, a normal shock. The shock quickly dissipated on the freestream surface as the flow expanded around the nose, creating an expansion wave which intersected and canceled the shock, although the shock affected the temperature and Mach number distribution on the waverider's surface. On the compression surface, the shock quickly became an oblique shock as the flow expanded around the nose, but as on the freestream surface, flow temperature and Mach number were affected, particularly in the nose region.

Figure 4-6 shows the temperature and Mach number variation over the entire freestream surface of the waverider. Figure 4-7 shows a zoom of the nose region. Note the rapid decrease in temperature away from the nose, and the somewhat slower, though still rapid rise in Mach number to the freestream value of Mach 10. The rise in Mach number is due to the combination of the increase in flow velocity and the drop in temperature.

Figure 4-8 shows the Mach number and temperature distributions on the whole of the compression surface, and Figure 4-9 shows a zoom of the nose region. The behavior of the flow on the compression surface is similar to that on the freestream surface, though on the compression surface the flow expands to match conditions behind an oblique shock. As on the freestream surface, the Mach number increases, and temperature rapidly decreases. On the compression surface, however, the temperature is generally higher than on the freestream surface, and the Mach number lower.

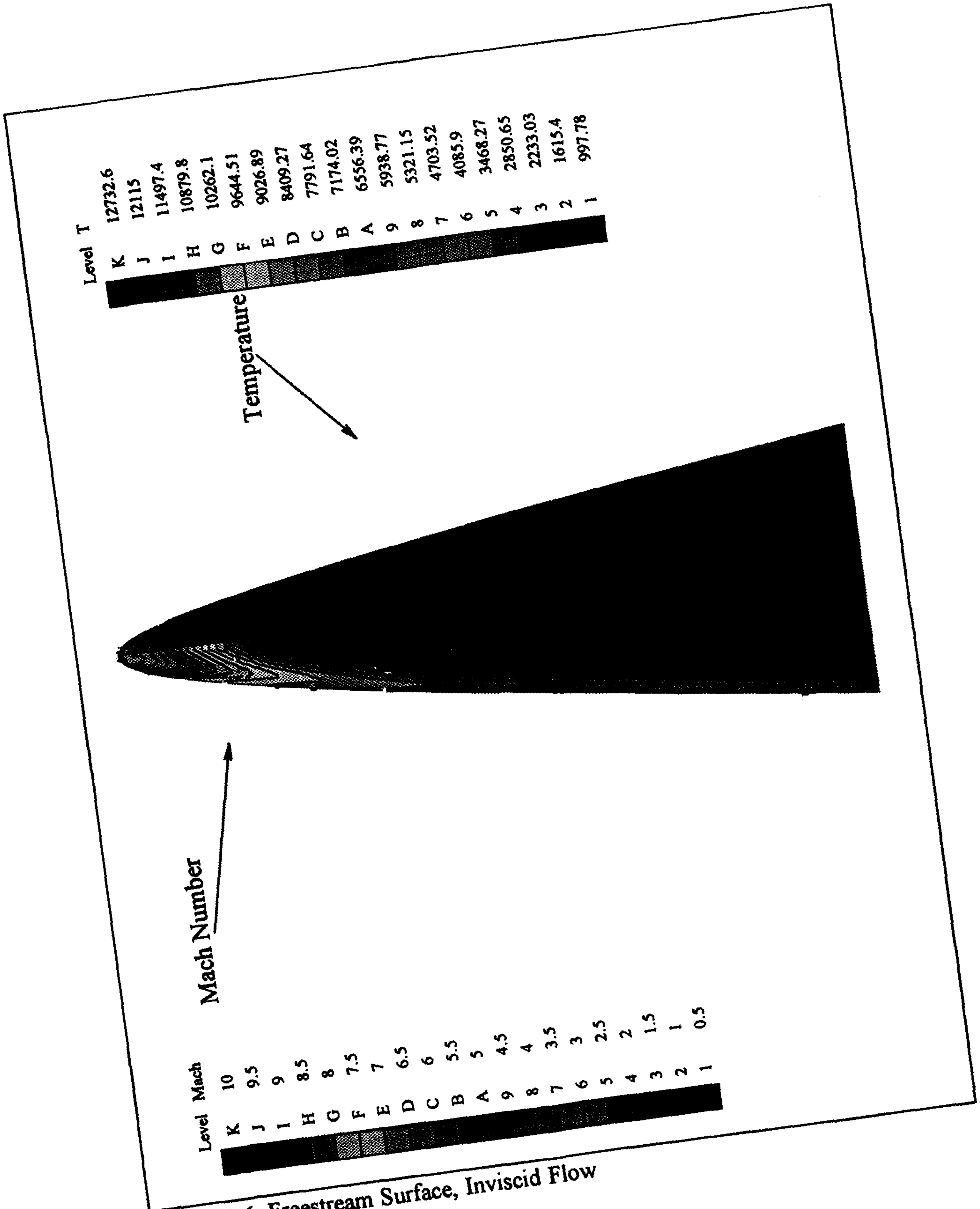

**Figure 4-6**, Freestream Surface, Inviscid Flow

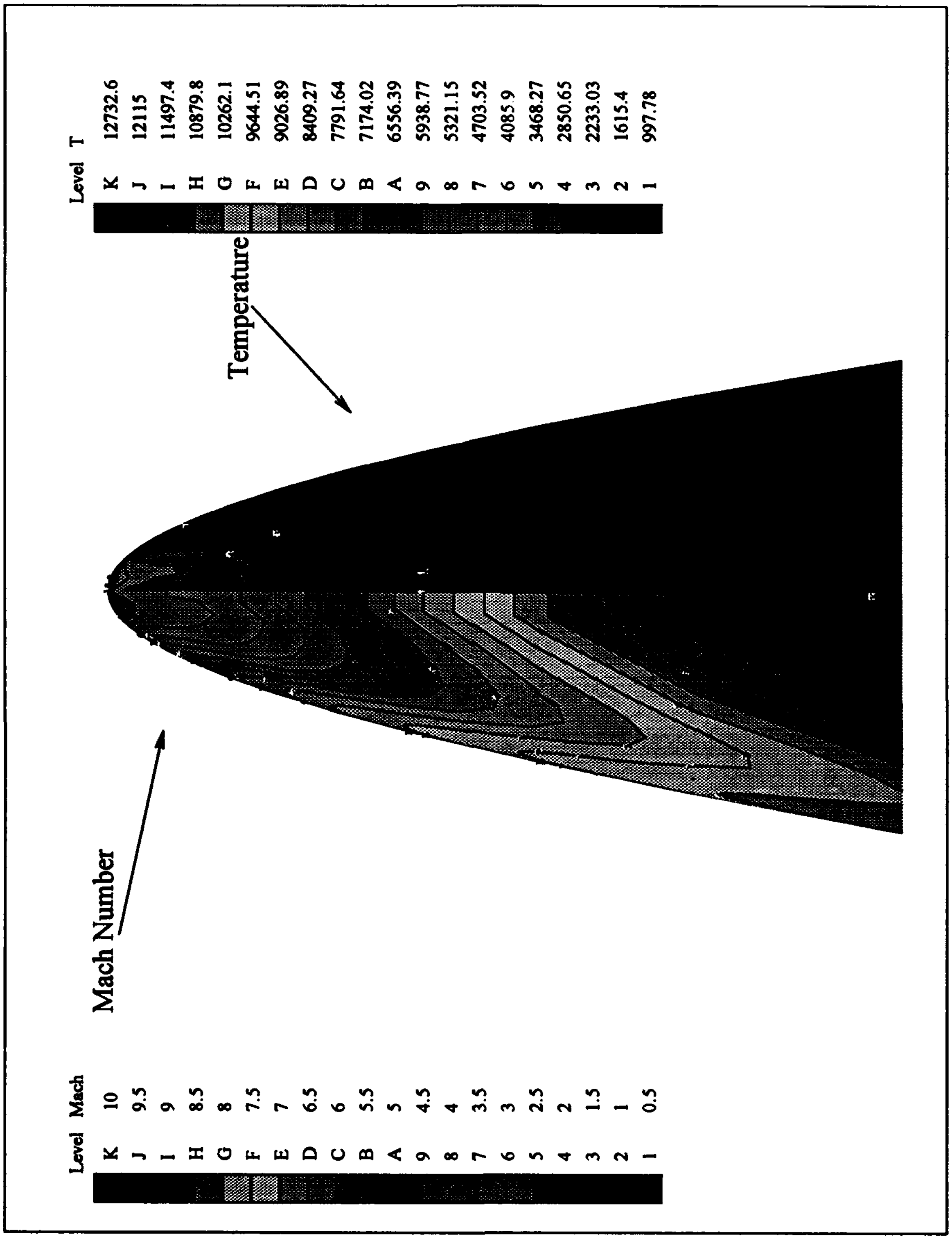

**Figure 4-7**, Zoom of Freestream Surface, Inviscid Flow

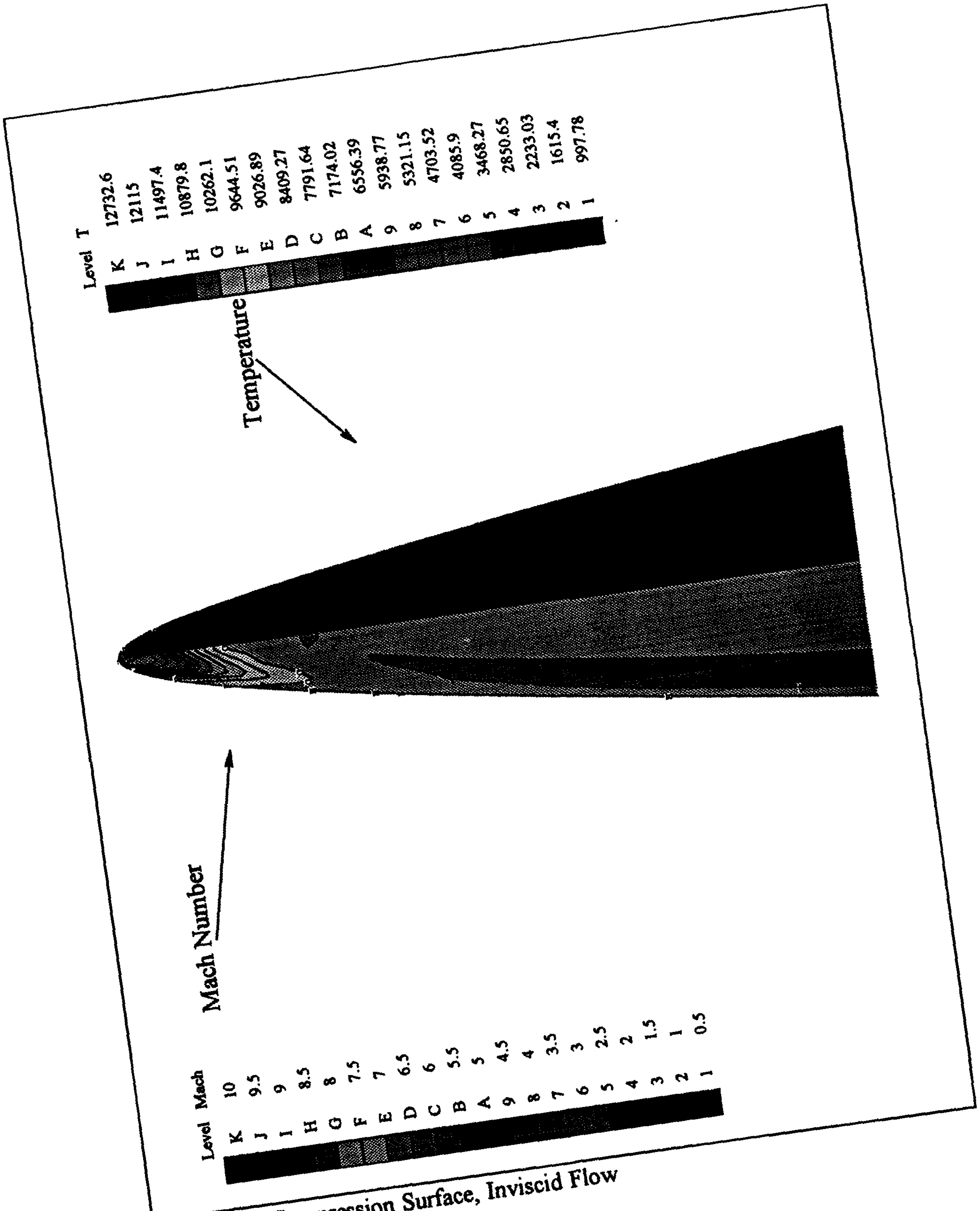

**Figure 4-8**, Compression Surface, Inviscid Flow

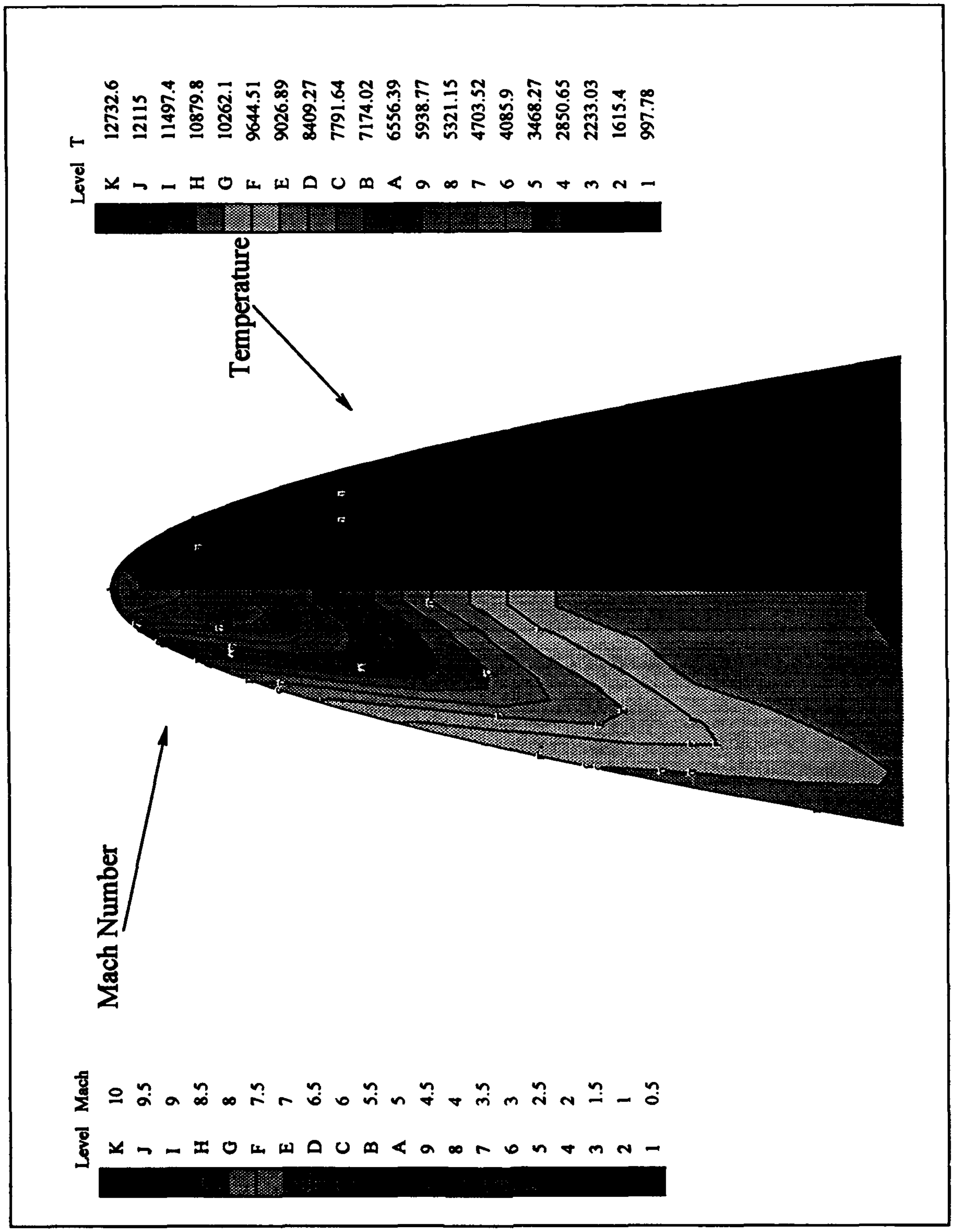

**Figure 4-9, Zoom of Compression Surface, Inviscid Flow**

## 4.2 Viscous Flow Results

Convergence for the Navier-Stokes equations and the computational grid required to resolve the flow was quite slow. Small grid volumes near the leading edges and nose, while providing good flow resolution, required very small values for the time step in order to keep from exceeding the allowable CFL stability criteria. Figure 4-10 shows the convergence history for the viscous flow case.

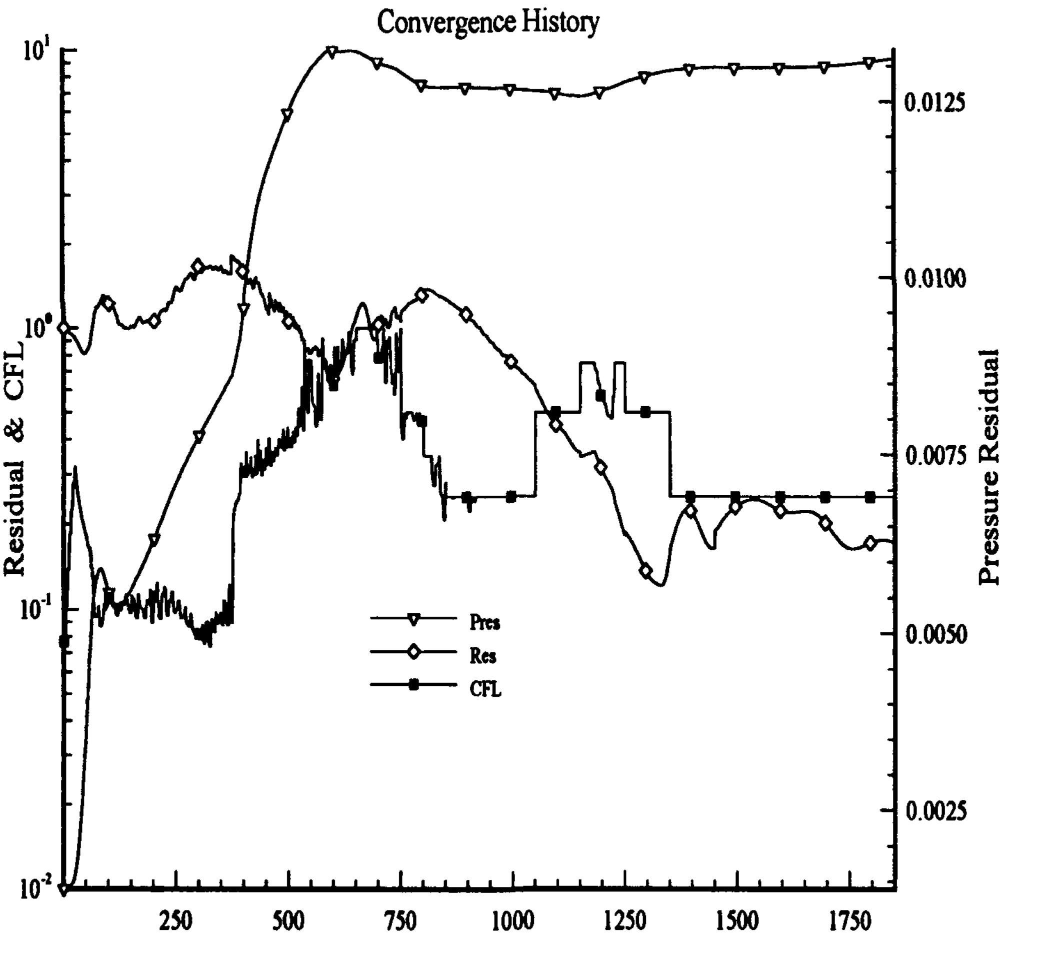

**Figure 4-10,** Viscous Flow Convergence History

The convergence went as expected, though not as quickly as desired. The residual at first rose and then began a short series of cycles in which the maximum residual decreased. After a significant decrease in residual another series of cycles began in which the maximum residual reached in the cycle decreased. The CFL stability criteria was adjusted automatically by the CFD code as necessary. At first the CFL number was quite low, on the order of 0.09, but then rose rapidly. The CFL number did not remain steady at the maximum allowed until approximately the 800th iteration. The root mean square surface pressure, or pressure residual, behaved as expected, rising rapidly and then leveling off. A slight decrease was noted, and, after a brief period of little to no change, the pressure residual began to rise slowly. A constant pressure residual is a good indication that the solution is converged, as the pressure on the waverider surface is not changing significantly, and thus the changes to the flow structure with each iteration are very minor.

As expected, when viscous terms are included in the equations, L/D drops due to the addition of viscous drag. The L/D ratio for the waverider in viscous flow is 5.74, a 30% drop from the inviscid case. This value of L/D is similar to that calculated in other solutions of viscous flow over a waverider. Another primary contribution to drag is wave drag, which is caused by the shock waves. In this case a weak shock formed over the freestream surface of the waverider, as illustrated by the lower part of Figure 4-11. This shock was formed in part by the bow shock produced by the blunt leading edges, and in part by the formation of the boundary layer. The shock on the compression surface was also affected by the formation of the boundary layer. The boundary layer on the compression surface was thinner than on the freestream surface, but had the effect of increasing the effective flow turning angle, increasing shock strength slightly. The stronger shocks, caused by the interaction of the boundary layer and shock waves, contributed to the drop in L/D as compared to the inviscid flow case by increasing the wave drag. Thus viscosity contributed to drag in two ways. It contributed directly and by increasing the strength of the shocks, and thus the wave drag. The computed L/D ratio compared favorably with results for similar vehicles. The computed L/D was greater than that determined by Müller (22), who arrived at a value of 4.18 for a waverider flying at Mach 5.5, though substantially less than those reported by Chang (6) and Takashima (31). Compared to Rasumussen's (26) analytic work, which gave results very like those reported by Müller, the computed value was high, though not unreasonably so. Comparing the result of 5.74 obtained in this research to a plot of L/D such as that in Anderson (5), this result can be seen to be quite near the predicted value, and the results obtained by others.

Figure 4-11 illustrates the pressure field at the waverider's line of symmetry. Figure 4-11 shows the oblique shock that forms below the compression surface (though, since the waverider is depicted herein upside down, the shock appears to be above the waverider), and the shock and expansion that form above the freestream surface. Figure 4-12 shows density contours for the same region as that depicted in Figure 4-11. The shock on the compression side is clearly depicted, as is the much weaker shock on the freestream side. Figure 4-13 is a zoom of the density contours in the nose region. This figure shows the shock on the freestream side more clearly than does Figure 4-12.

Figure 4-14 shows the Mach number contours in the nose region. The expansion can be seen here as, just above the freestream surface, the flow accelerates. Note also the rapid increase in Mach number above the surface, indicating how thin the boundary layer is in the nose region. Figure 4-15 is a further zoom of the Mach contours in the region of the waverider's nose. This figure shows even better than the previous one how thin the boundary layer is in this region. It also shows the shock structure in the nose region quite clearly.

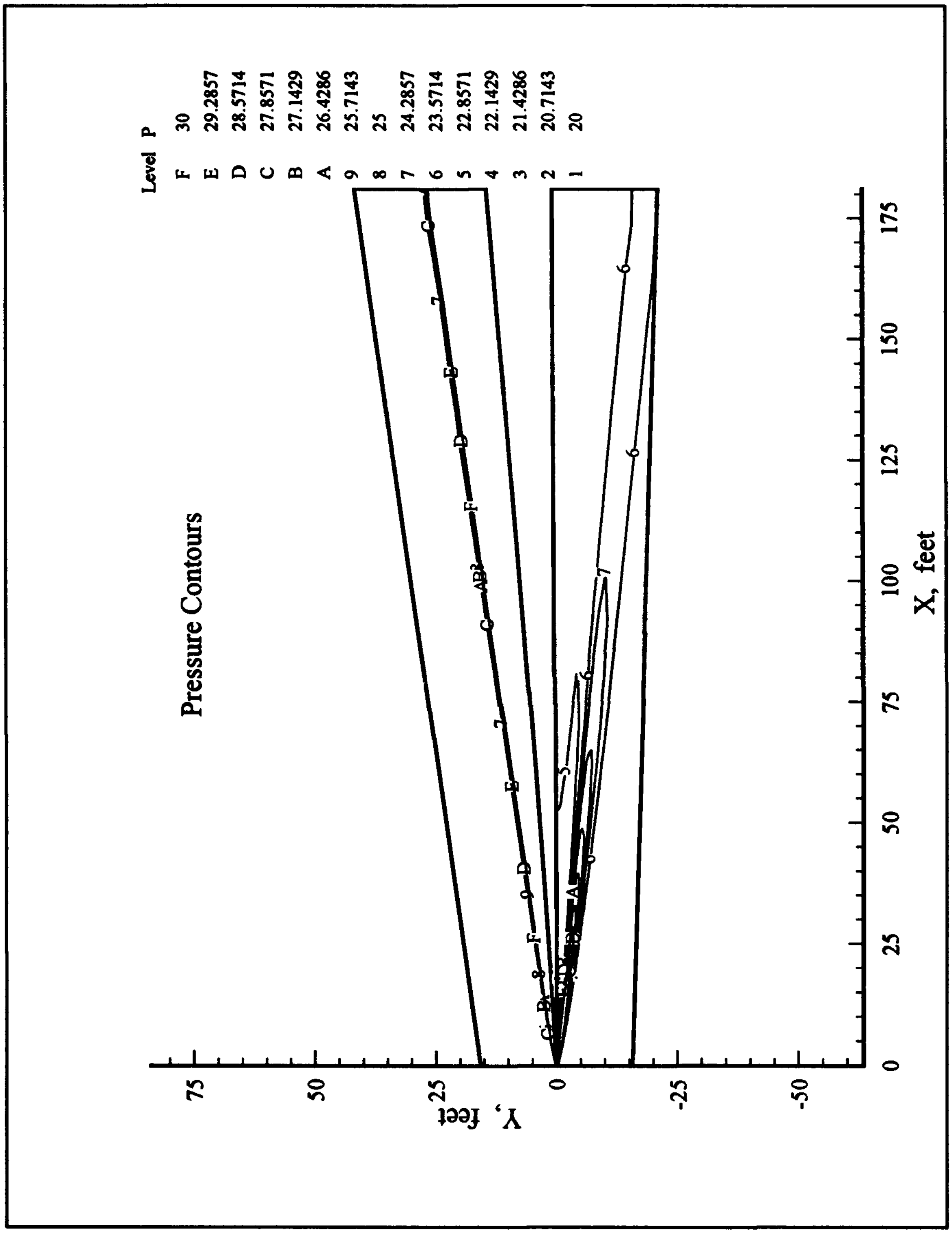

**Figure 4-11**, Centerline Pressure Contours

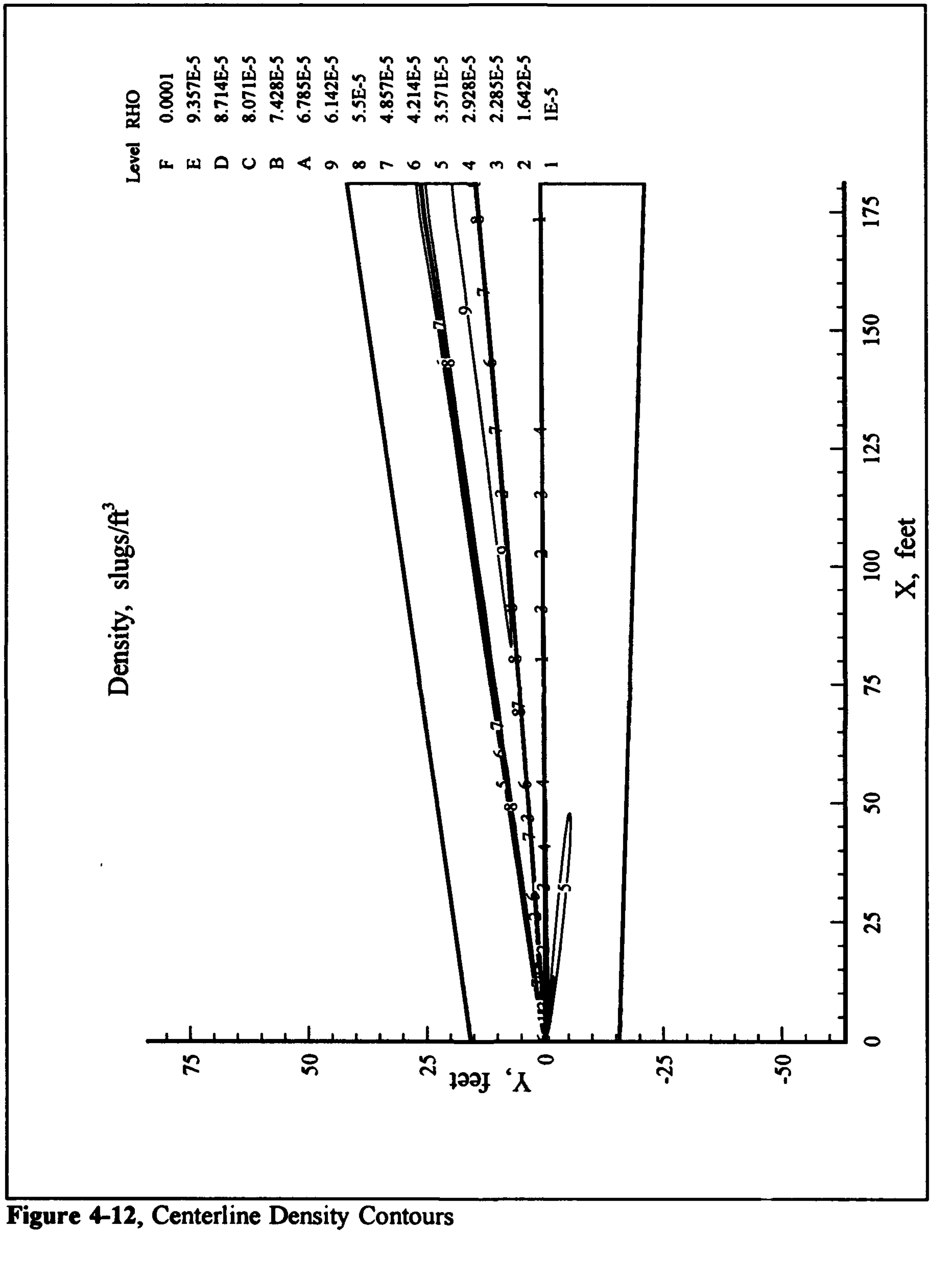

Figure 4-12, Centerline Density Contours

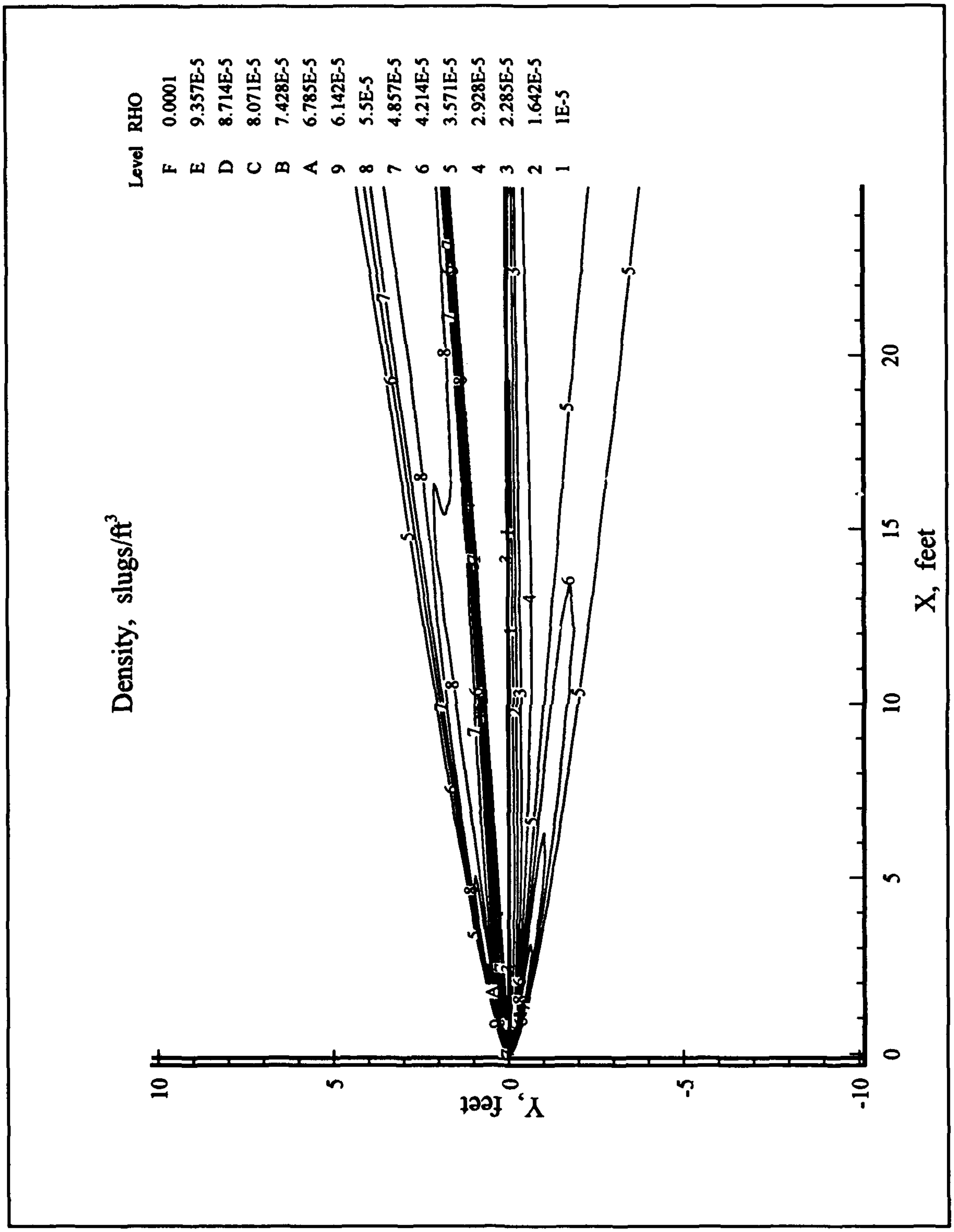

**Figure 4-13**, Zoom of Nose Region, Centerline Density Contours

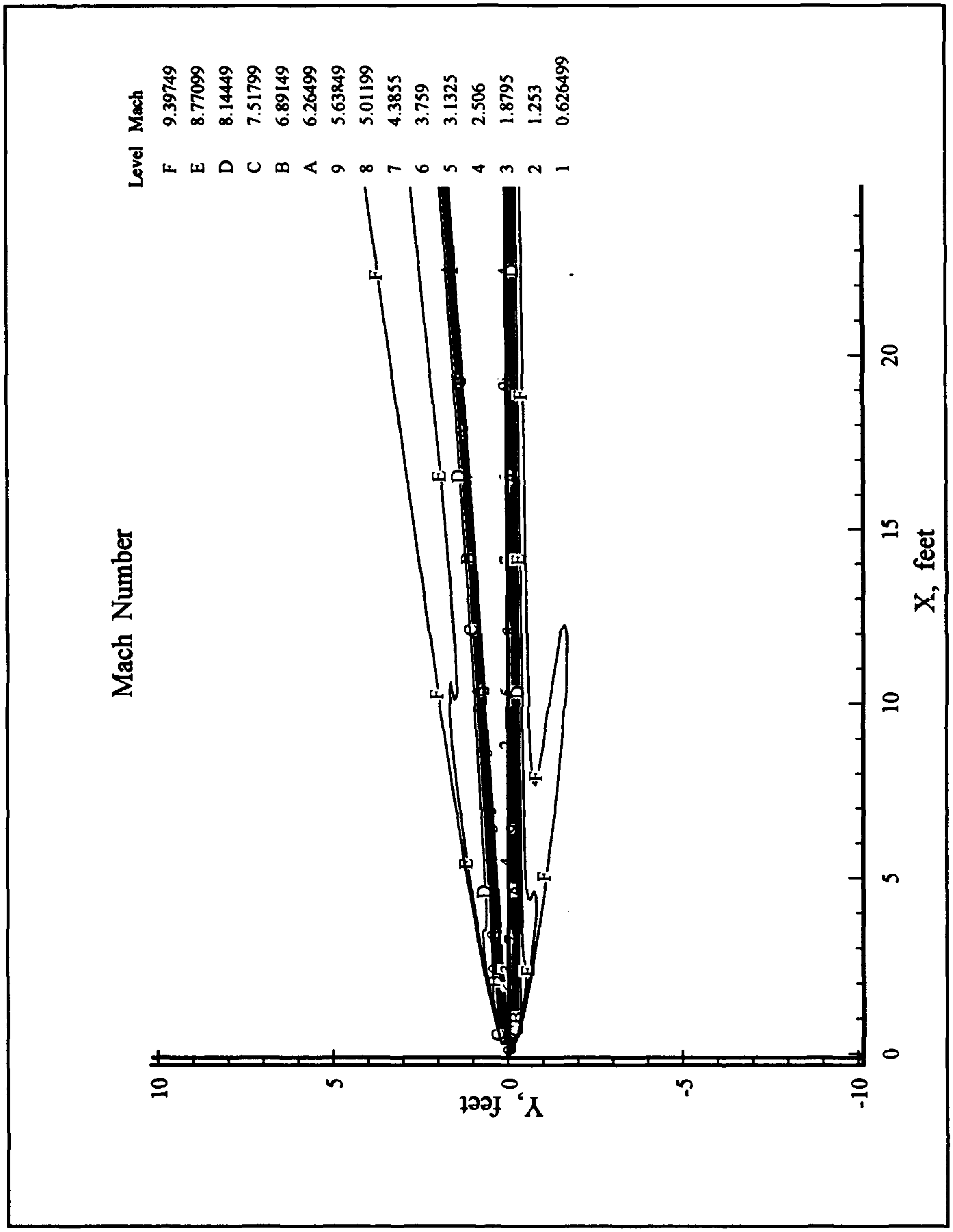

**Figure 4-14**, Centerline Mach Contours

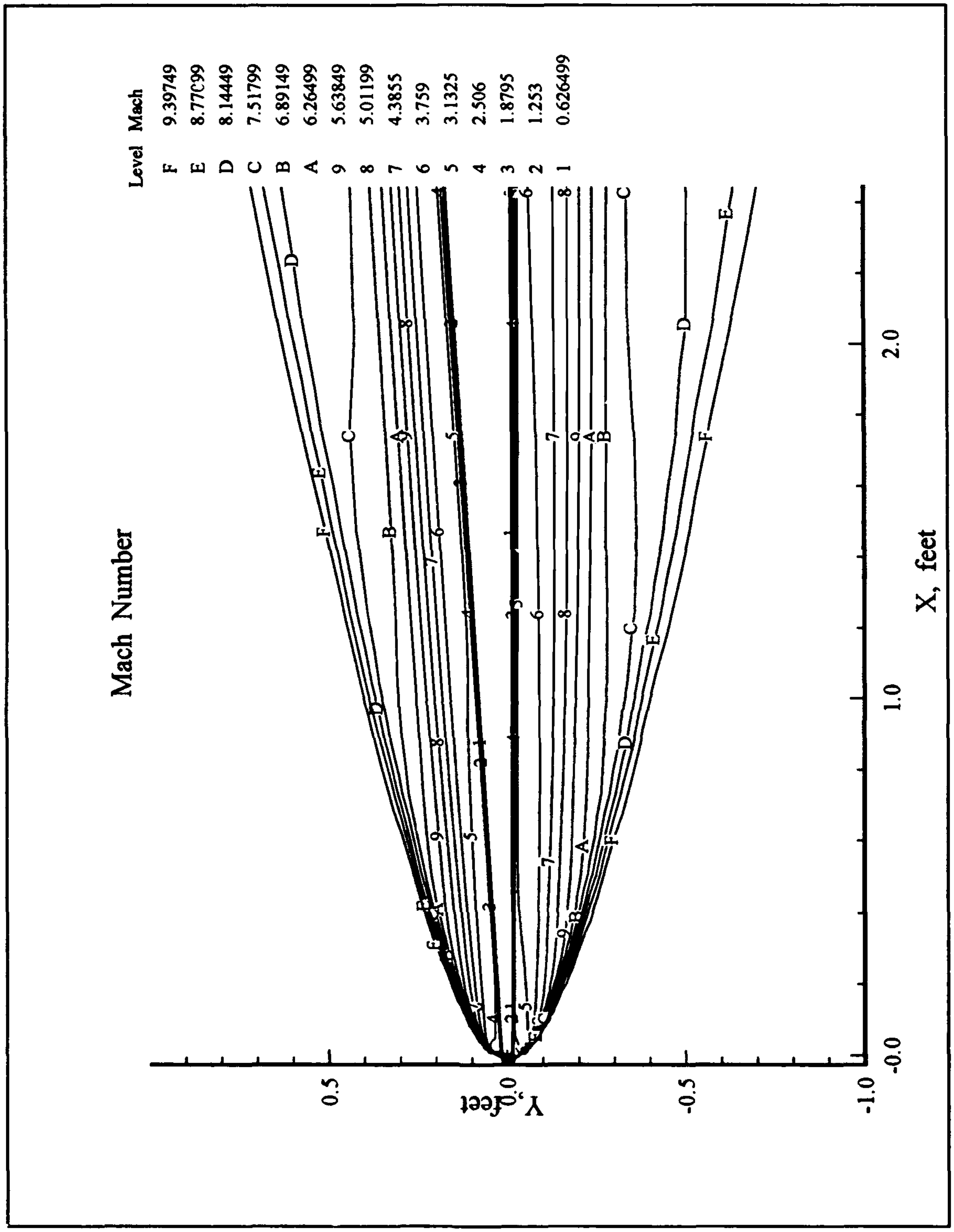

**Figure 4-15**, Zoom of Nose Region, Centerline Mach Contours

The velocity vector plots, Figure 4-16 through Figure 4-18, show the steep velocity gradients in the boundary layer quite well. The boundary layer grew more quickly on the freestream surface than on the compression surface due to the lower pressures. Comparing Figure 4-17 with Figure 4-18 it can be seen that, at the waverider's trailing edge, the boundary layer on the freestream surface is almost twice as thick as on the compression surface.

The temperature, as shown in the plots of temperature and density vs. Y (Figure 4-19 through Figure 4-22), changes as predicted by theory for a cold wall; climbing deeper into the boundary layer to a maximum just above the surface, and then falling off sharply to the surface temperature. In the plots of temperature presented here, the temperature did not reach the surface temperature as the data was calculated at the first cell center, which was positioned just above the surface of the waverider. Interpolation between the value at the first point above the waverider surface and the ghost point just below the surface returns the value for temperature imposed as a boundary condition. Figure 4-19 shows more the effects of the shock and expansion than the boundary layer, due to the boundary layer's thinness near the nose. Figure 4-20 also shows more the effects of the expansion than the boundary layer. The sudden turn in the plot at Y = 0.025 feet is due to the thinness of the boundary layer. The boundary layer is entirely missed, and the flow almost jumps to the imposed boundary conditions.

The plots of flow variables in the boundary layer near the nose region show how thin the boundary layer is there. Also, because the shock angle is so small, the plots extend almost to the shock. A much finer grid would be required to resolve the boundary layer at all near the waverider nose.

The plots of temperature and density near the trailing edge, Figure 4-21 and Figure 4-22, show the expected behavior, described above, very well. Both plots show temperature rising sharply just above the waverider surface to a maximum that is approximately at the center of the boundary layer, and then falling off to the temperature at the edge of the boundary layer. Density mirrored the behavior of temperature, indicating an almost constant pressure through the boundary layer.

The plots of velocity and Mach number, Figure 4-23 through Figure 4-26, mirror the behavior seen in the vector plots, Figure 4-17 and Figure 4-18. As the wall is approached the velocity drops rapidly to zero. As explained above, the values shown don't actually reach zero velocity, as the values are not calculated on the surface of the waverider, but rather, just above it. Interpolation between the ghost point and the first point above the waverider surface shows that both the velocity and Mach number are zero at the surface. The plots of velocity and Mach number near the nose, Figure 4-24 and Figure 4-23 show the effects more of the flow expansion in that region than the boundary layer. The plot of the velocity profile above the compression surface shown in Figure 4-23 looks like a boundary layer profile, but is actually caused by the flow expansion in the region. The plot of the velocity profile above the freestream surface, Figure 4-24, shows a very thin boundary layer. Like Figure 4-20, the values "jump", that is the slope is discontinuous, at the points just above the waverider surface. This is due to inadequate resolution of the flow in the boundary layer, and the boundary conditions being imposed, just as explained above for the plots of temperature and density.

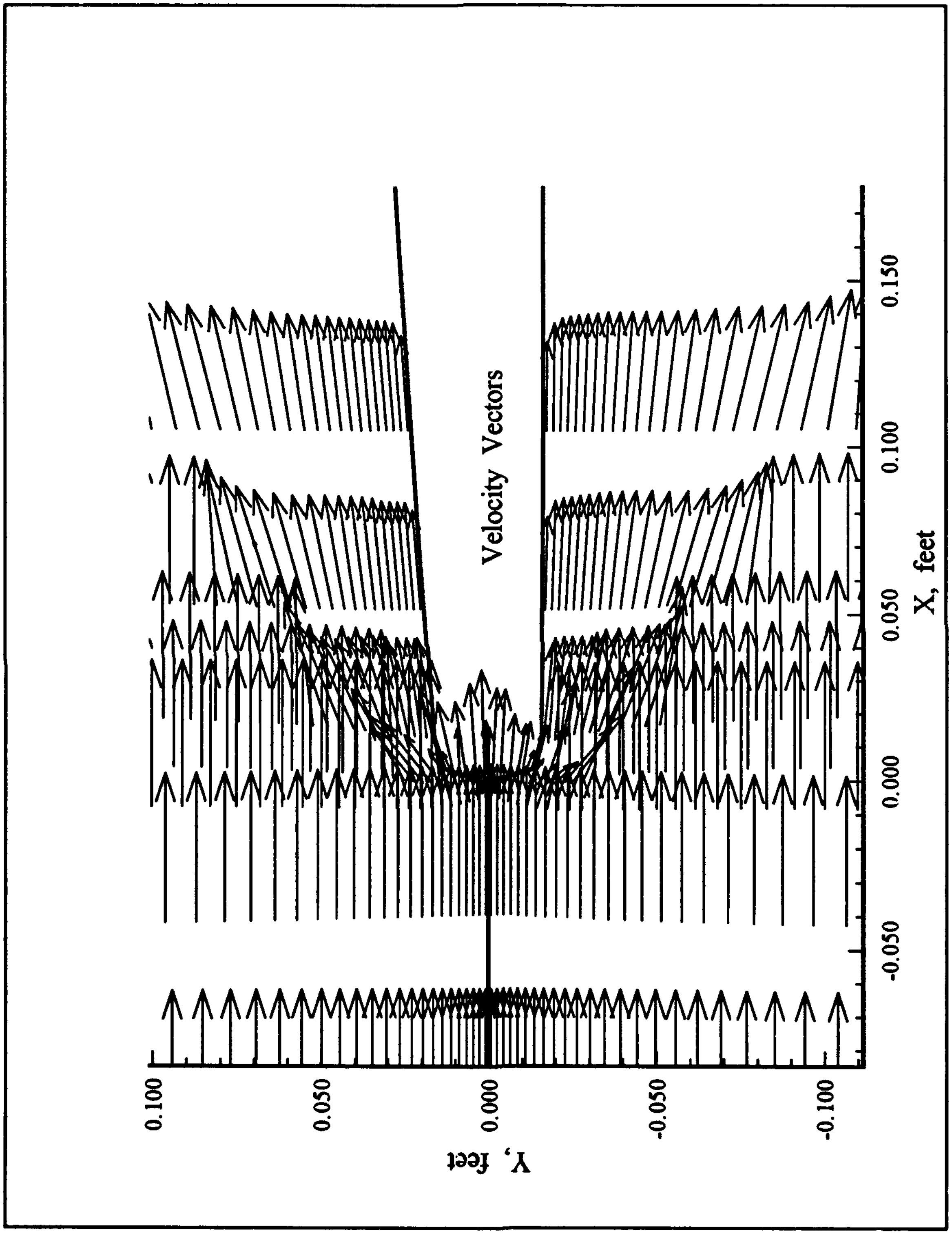

**Figure 4-16**, Centerline Velocity Vector Plot, Nose Region

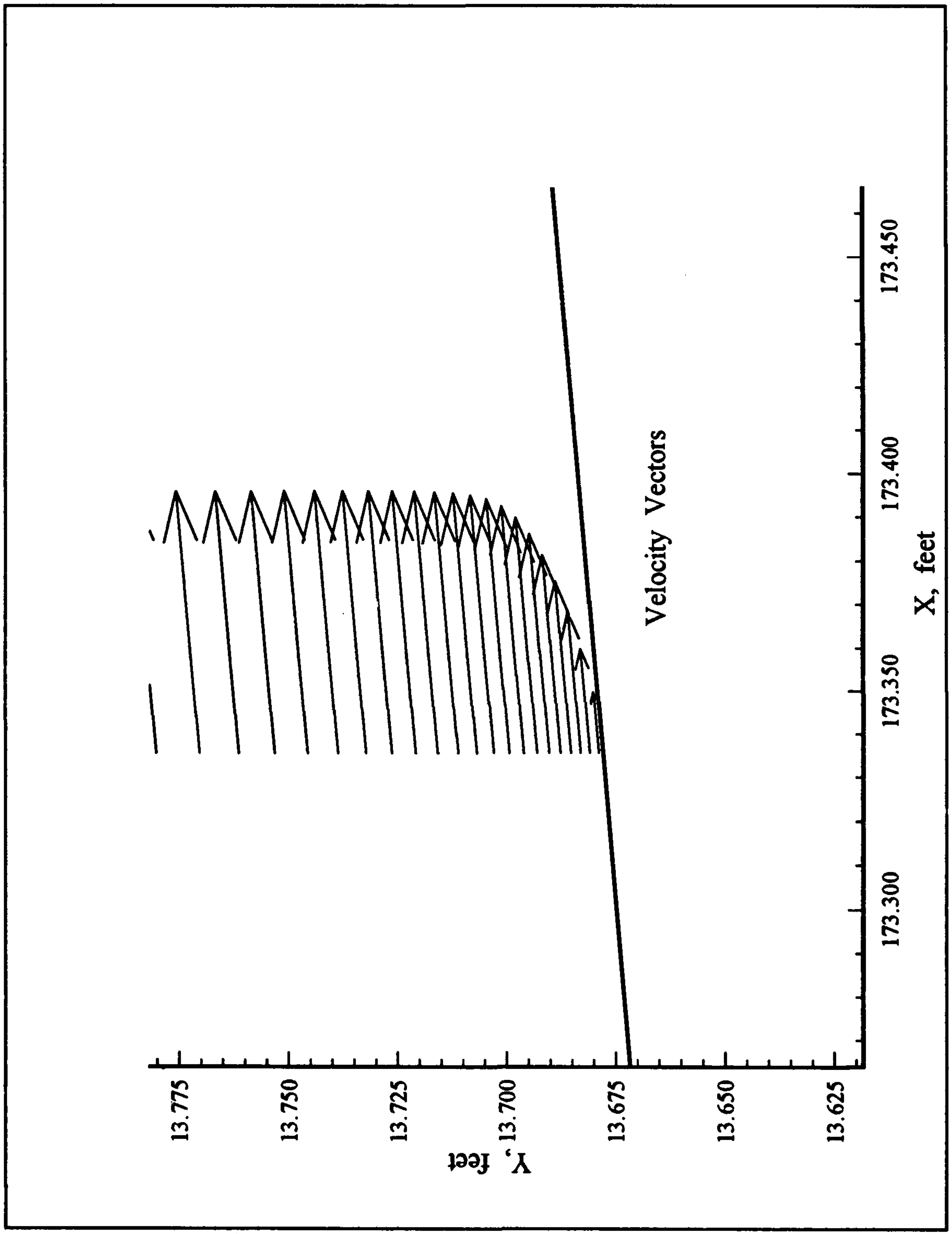

**Figure 4-17**, Centerline Velocity Vector Plot, Trailing Edge, Compression Surface

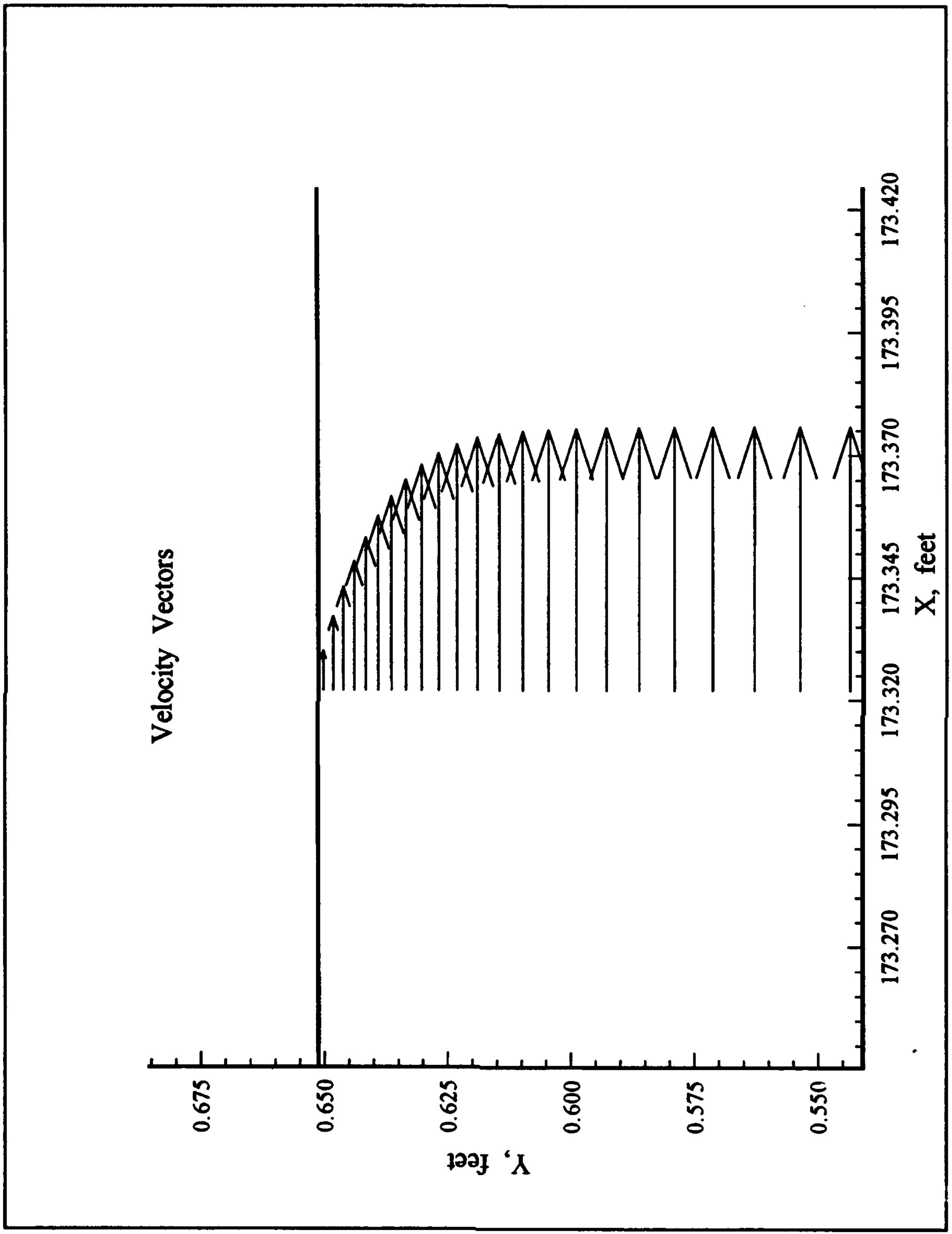

**Figure 4-18**, Centerline Velocity Vector Plot, Trailing Edge, Freestream Surface

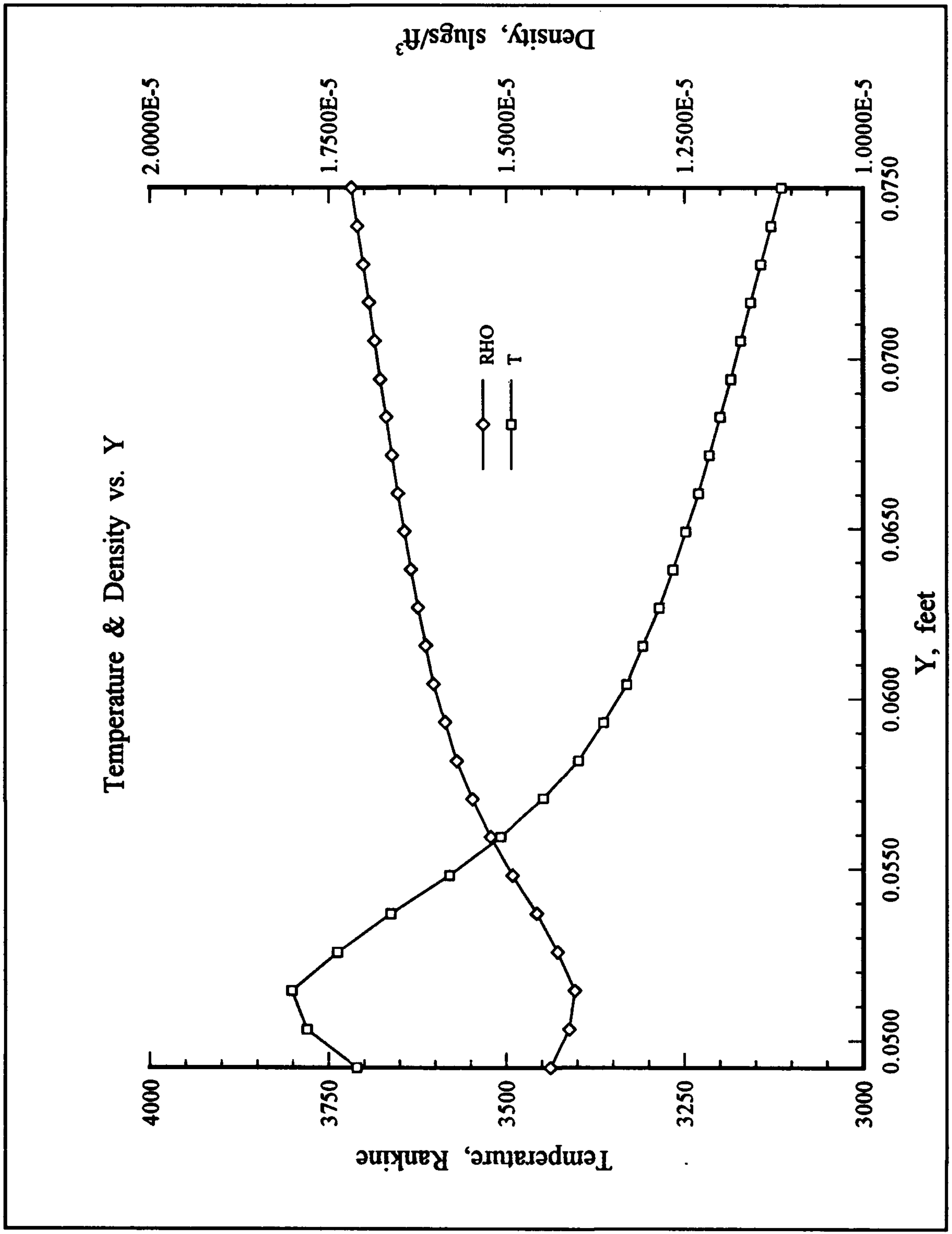

**Figure 4-19**, Compression Surface Centerline Boundary Layer Data, Nose Region

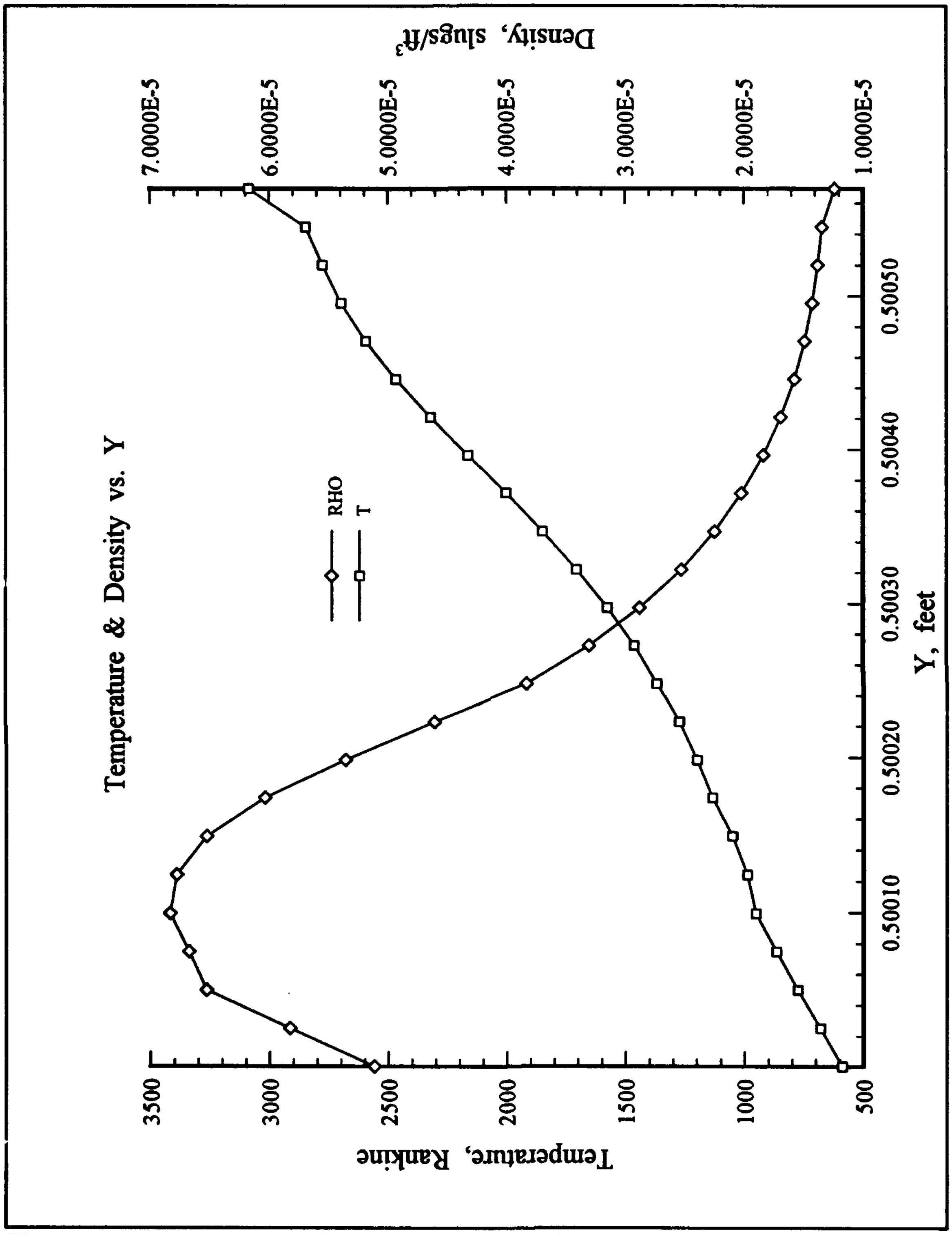

**Figure 4-20**, Freestream Surface Centerline Boundary Layer Data, Nose Region

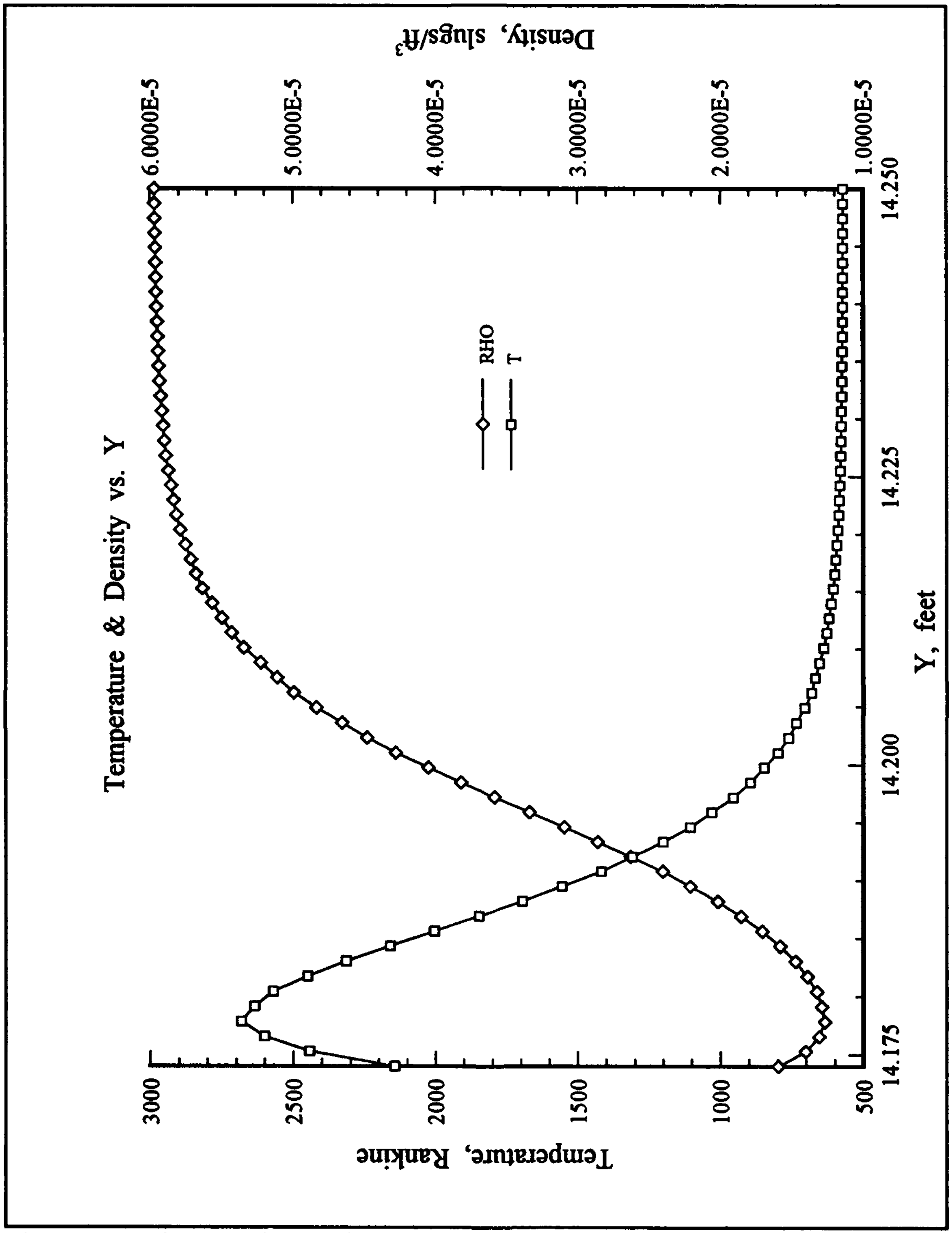

**Figure 4-21**, Compression Surface Centerline Boundary Layer Data, Trailing Edge

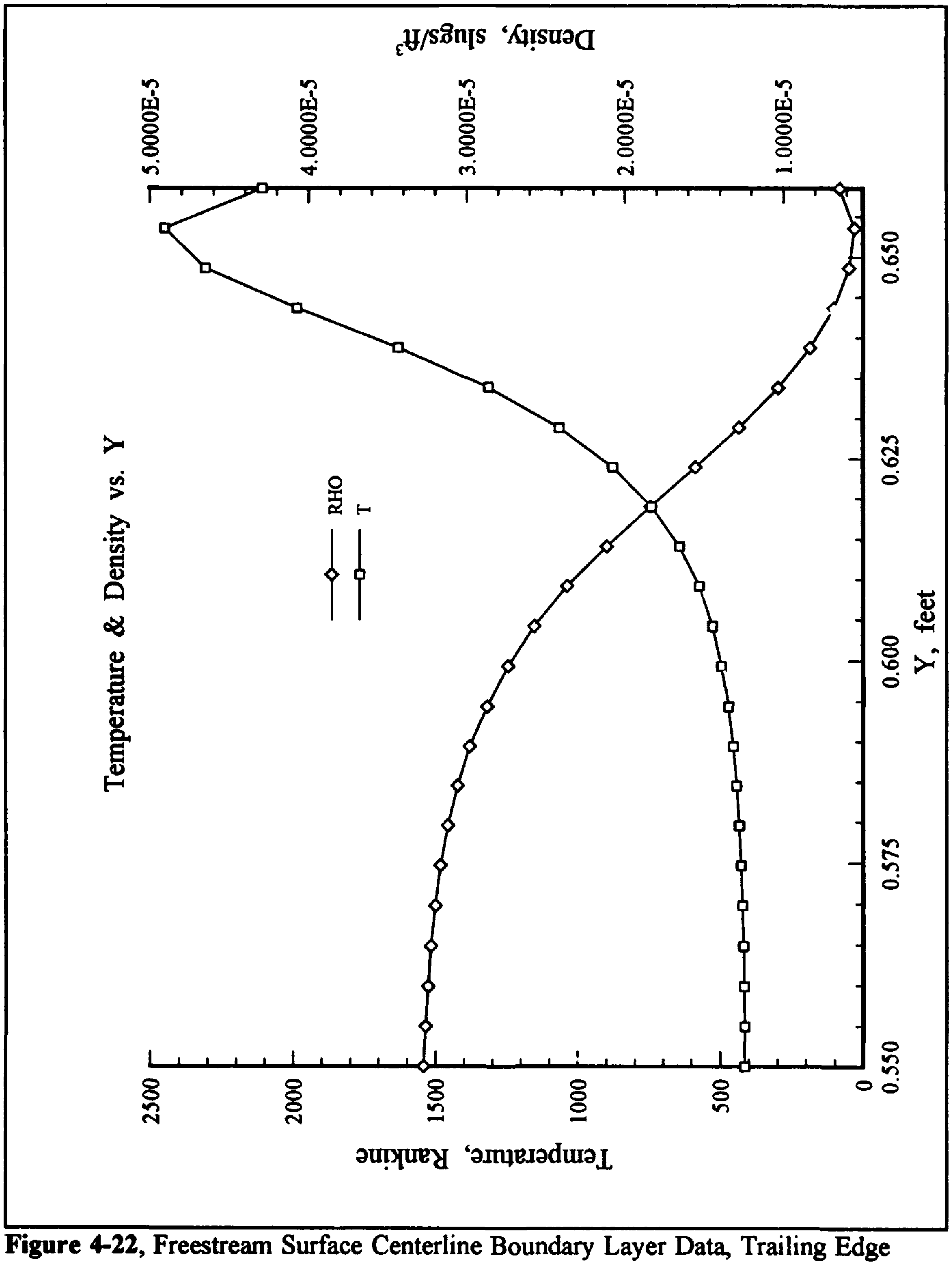

**Figure 4-22**, Freestream Surface Centerline Boundary Layer Data, Trailing Edge

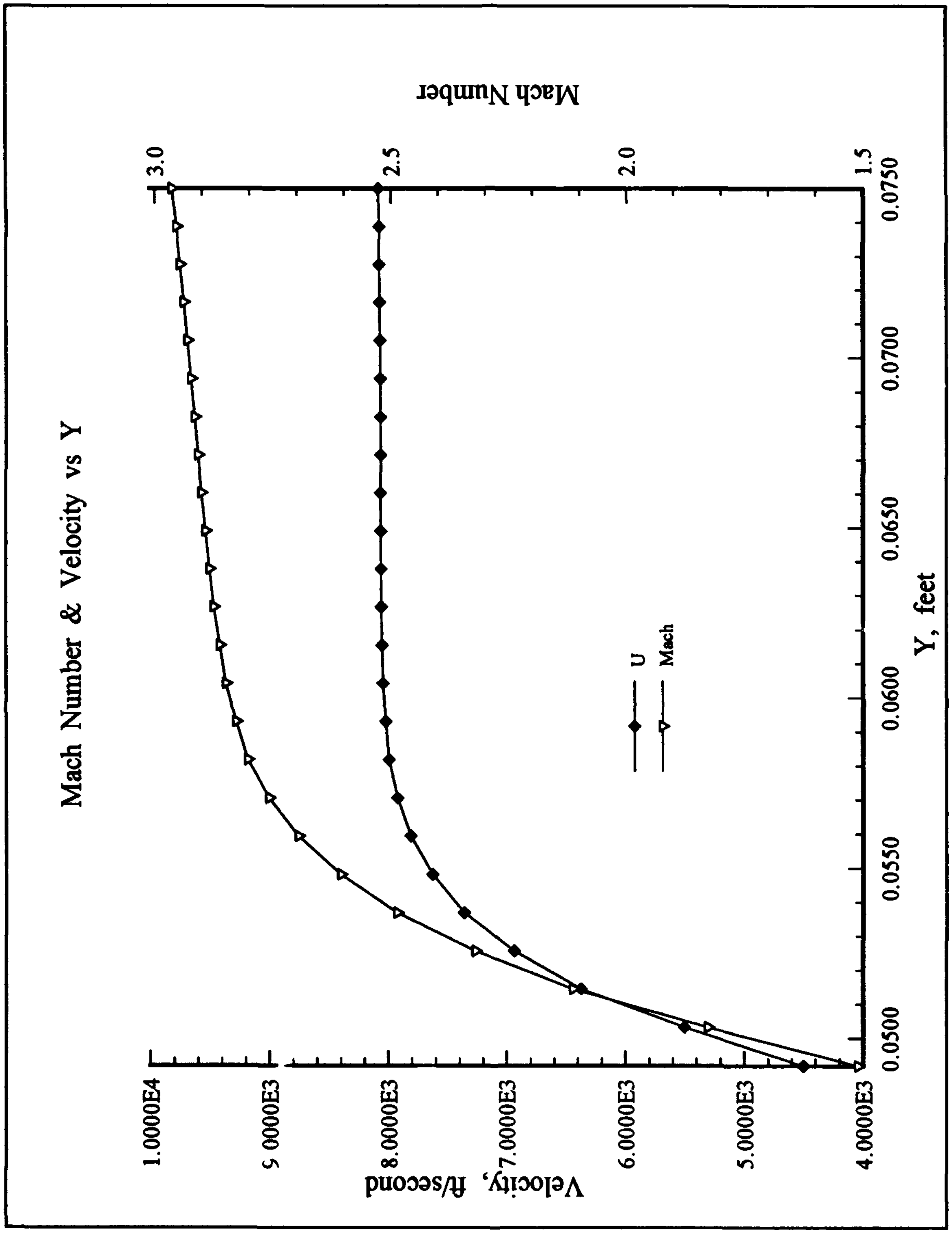

**Figure 4-23**, Compression Surface Centerline Boundary Layer Data, Nose Region

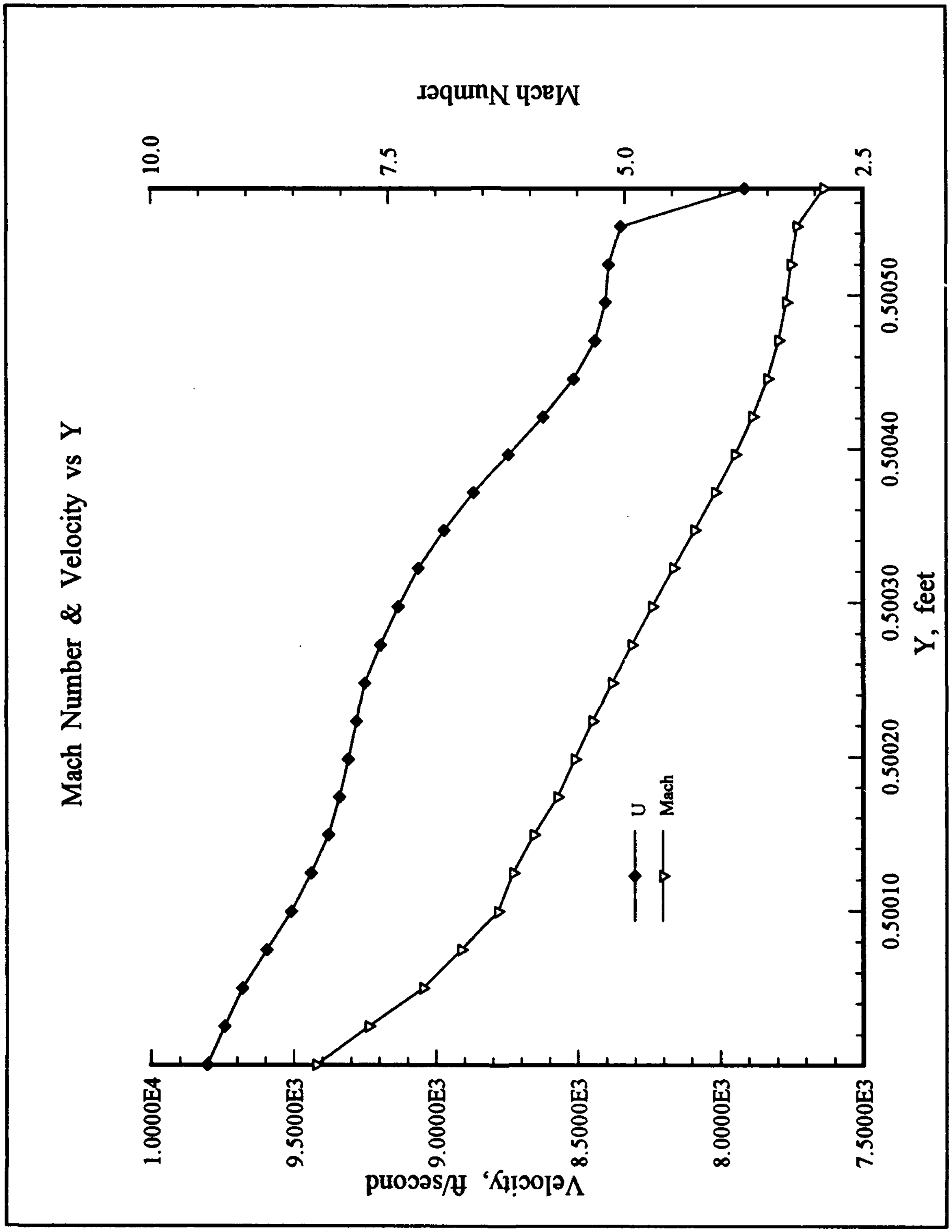

**Figure 4-24**, Freestream Surface Centerline Boundary Layer Data, Nose Region

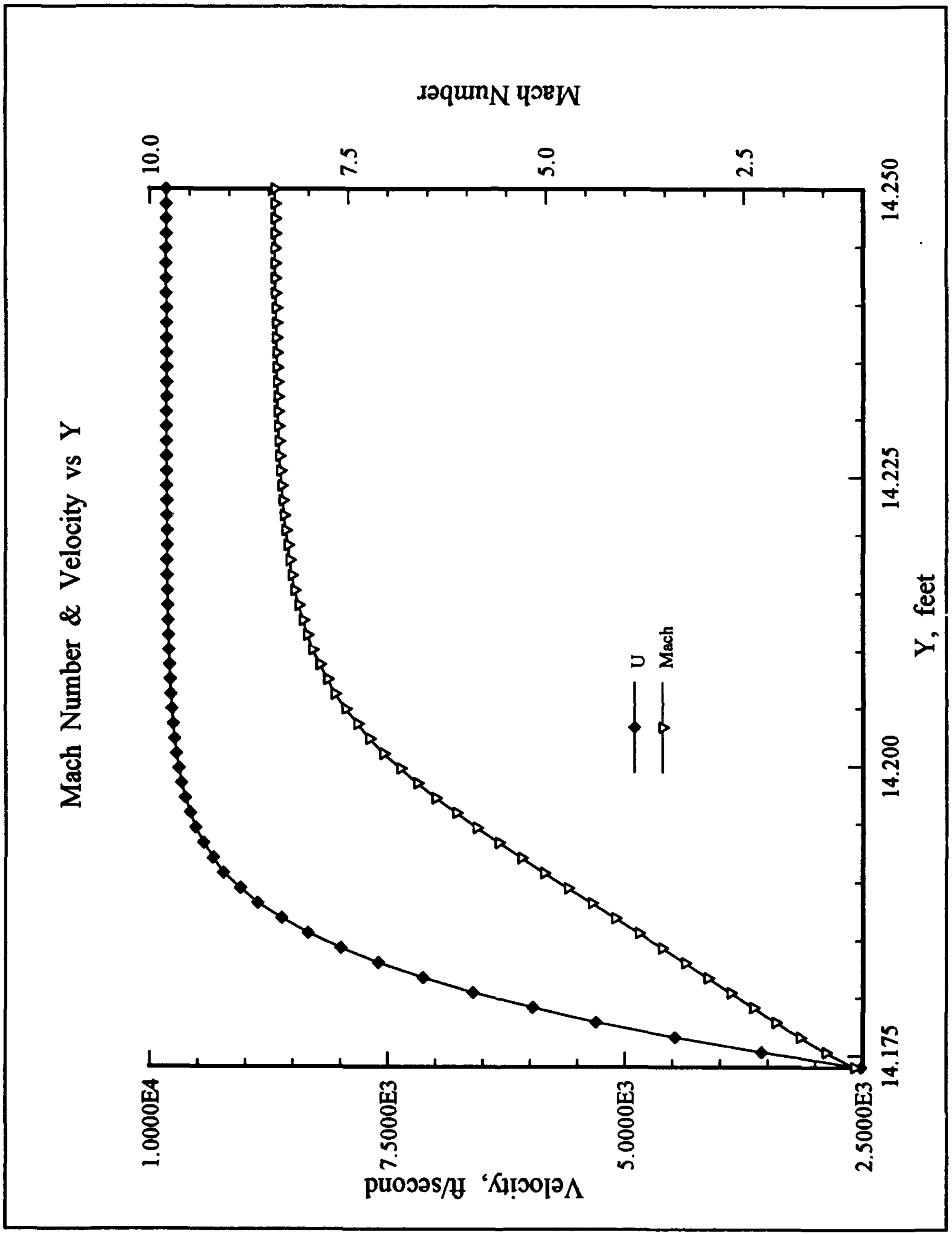

**Figure 4-25**, Compression Surface Centerline Boundary Layer Data, Trailing Edge

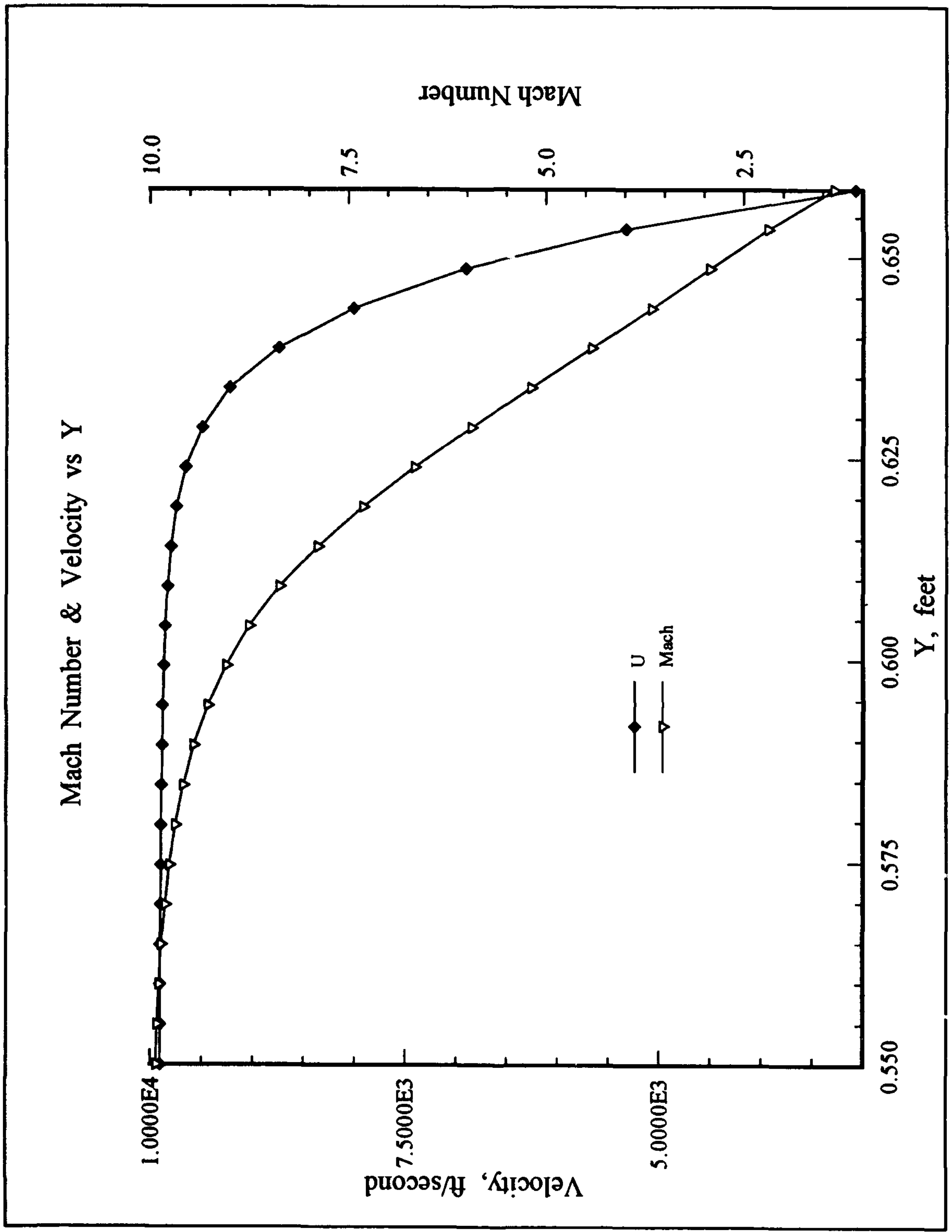

**Figure 4-26**, Freestream Surface Centerline Boundary Layer Data, Trailing Edge

Heat transfer from the flow to the waverider was greatest at the nose and leading edges, as was the skin friction coefficient. Figure 4-27 and Figure 4-28 illustrate the heat transfer and skin friction compression surface and Figure 4-29 and Figure 4-30 show these values on the freestream surface. The variation of heat transfer and skin friction was greatest in the spanwise direction, as can be seen in the following figures. The important thing to note in these figures is, as mentioned above, that the skin friction and heat transfer are greatest at the nose and leading edges.

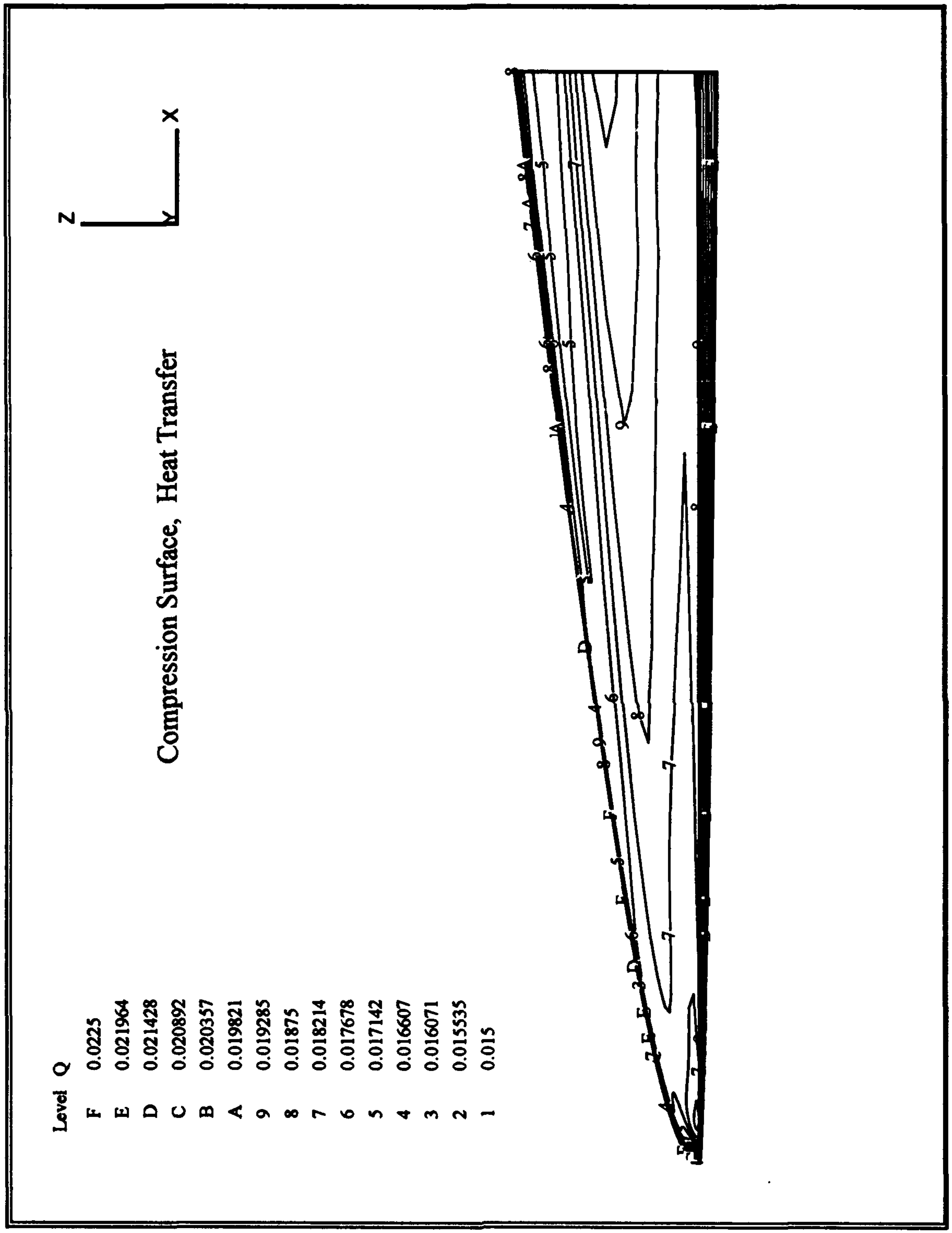

Figure 4-27, Compression Surface Stanton Number Distribution

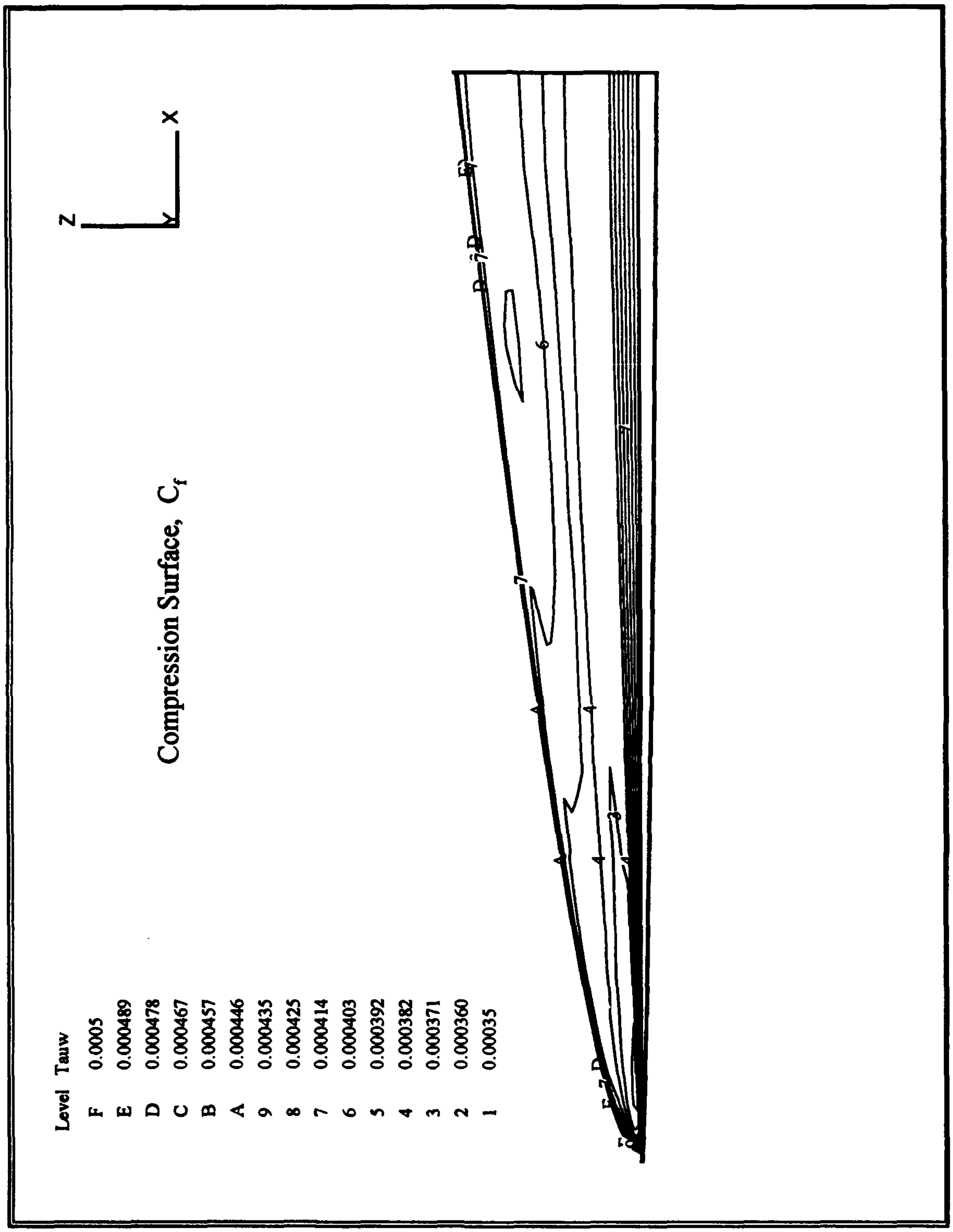

**Figure 4-28,** Compression Surface Skin Friction Coefficient Distribution

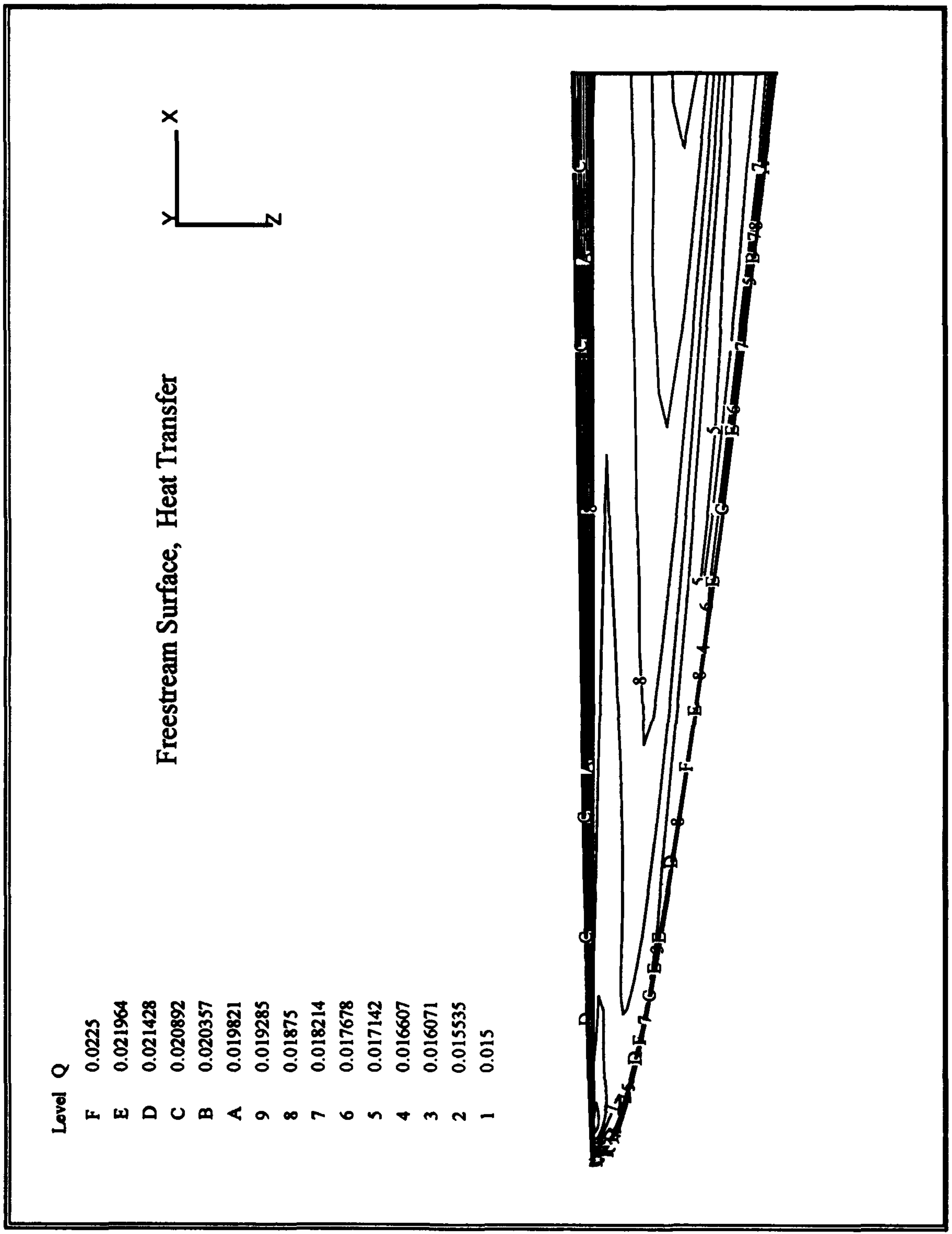

**Figure 4-29,** Freestream Surface Heat Transfer Distribution

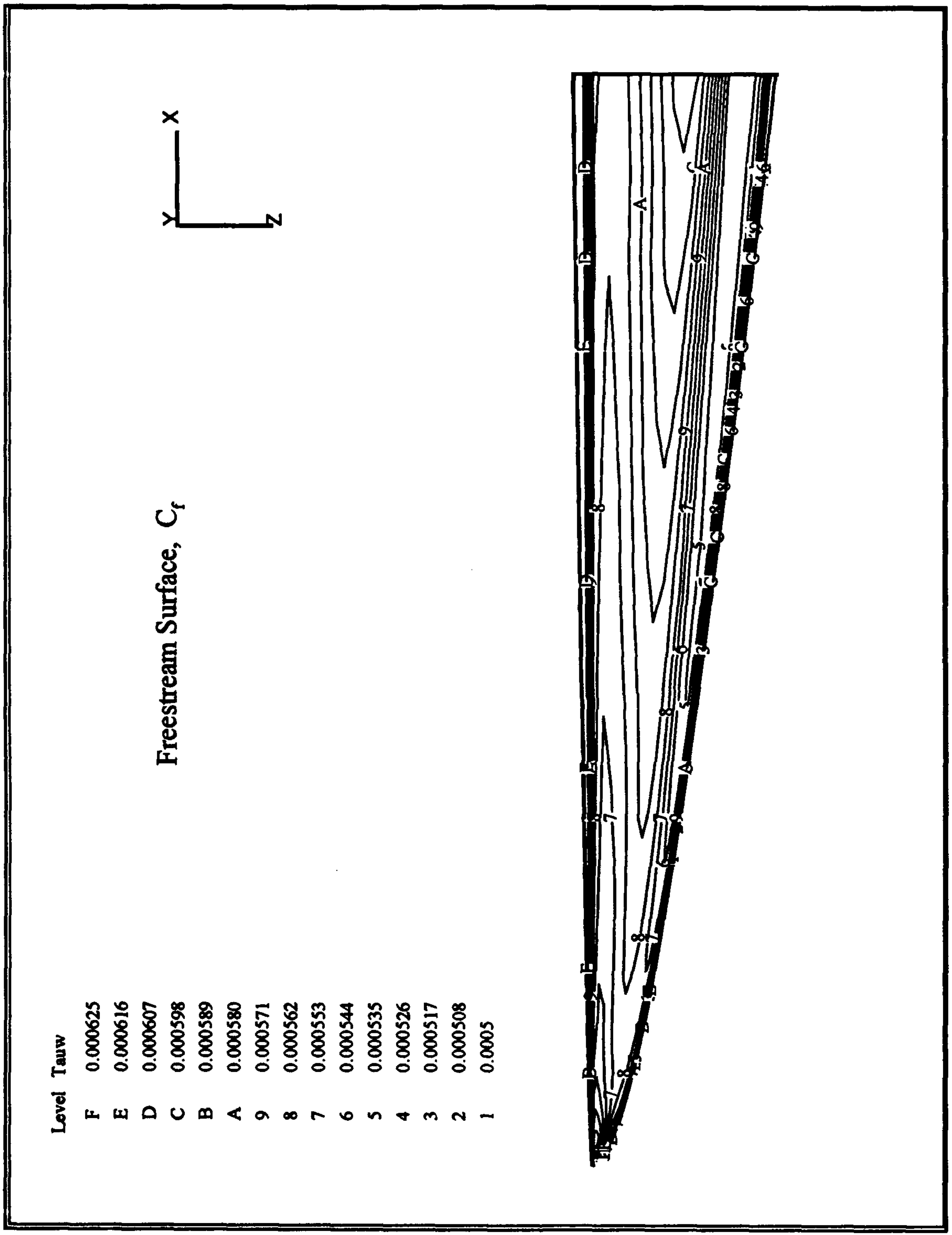

**Figure 4-30,** Freestream Surface Skin Friction Coefficient Distribution

Figure 4-31 illustrates the variation of pressure, temperature, and velocity on the stagnation stream line. The shock standoff from the nose is approximately 0.02 inches, a very small distance, indicating that the stagnation region will probably comprise a significant portion of this distance. As shown by Figure 4-31, the flow was not well resolved in the stagnation region. A much finer grid would be required to accurately resolve the flow in the stagnation region.

The figure does show, by inference, the extreme flow gradients present in the stagnation region. The flow will have to slow from a little more than 5000 feet per second to zero velocity within .025 feet. This will cause a corresponding rise in temperature as the kinetic energy is transformed to internal energy. Temperature will show extremely steep gradients, as it not only reaches a maximum, but then has to drop to match the wall temperature. These extreme gradients, combined with too coarse of a grid, make the numerical solution in the stagnation region to be of questionable value and accuracy.

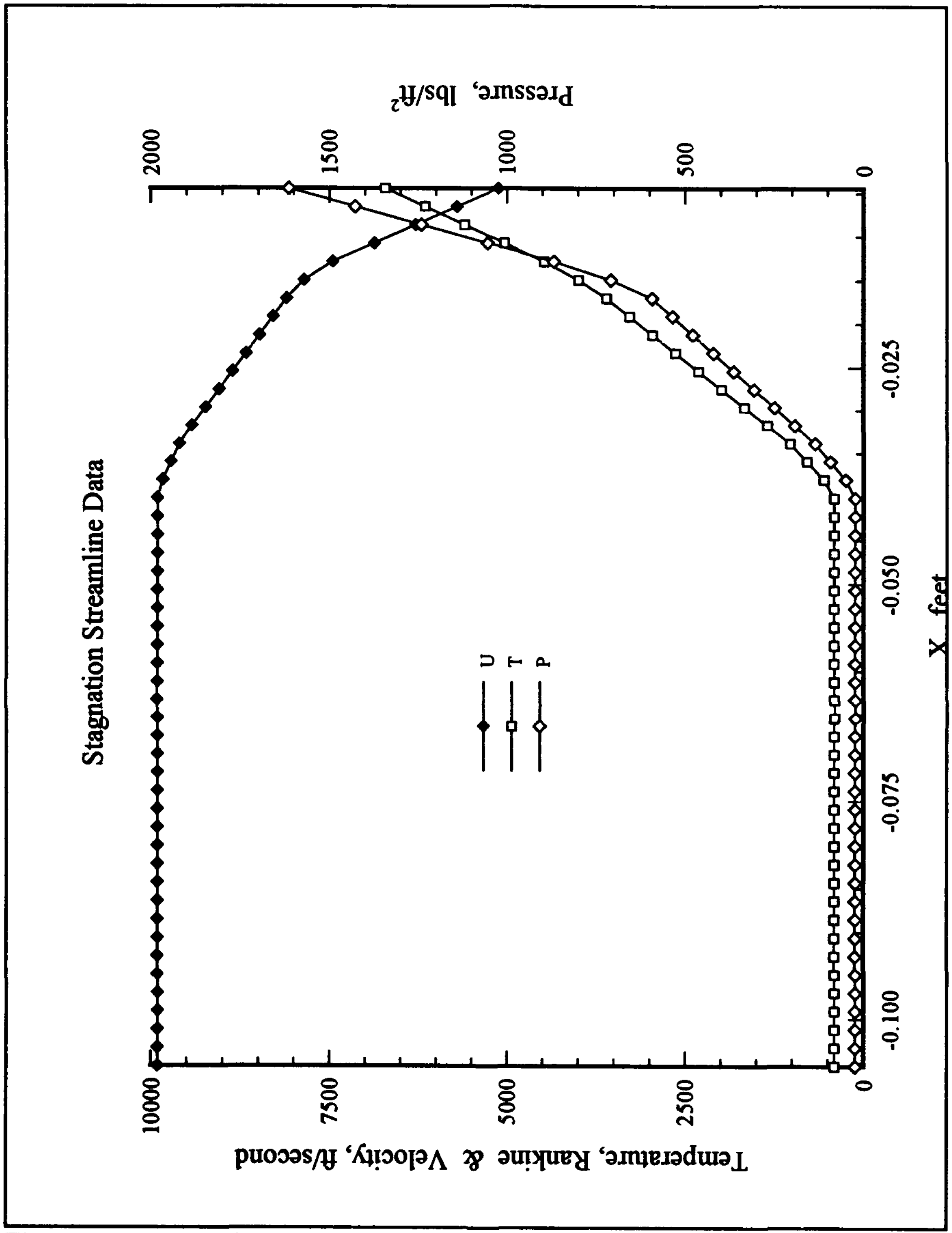

**Figure 4-31**, Stagnation Stream Line Flow Data

A Y-Z cross section of the flow near the baseplane, Figure 4-32, looked quite similar to those of the inviscid case, as shown in Figure 4-3 and Figure 4-4. The main differences between the viscous and inviscid cases included the boundary layer, a compression above the freestream surface due to the boundary layer, and the shock detaching from the leading edges of the waverider to stand off a short distance. The baseplane showed an interesting phenomena: a lambda shaped shock, shown in Figure 4-33. This phenomenon was caused by a flow expansion around the waverider's leading edge meeting the high pressure region on the compression surface. The strong compression from the low pressure just inboard of the leading edge to the pressure on the main part of the compression surface caused the shock to form. This shock structure did not exist two planes upstream of the baseplane; a single detached shock appears, as pictured in Figure 4-34, which is two planes upstream of the baseplane.

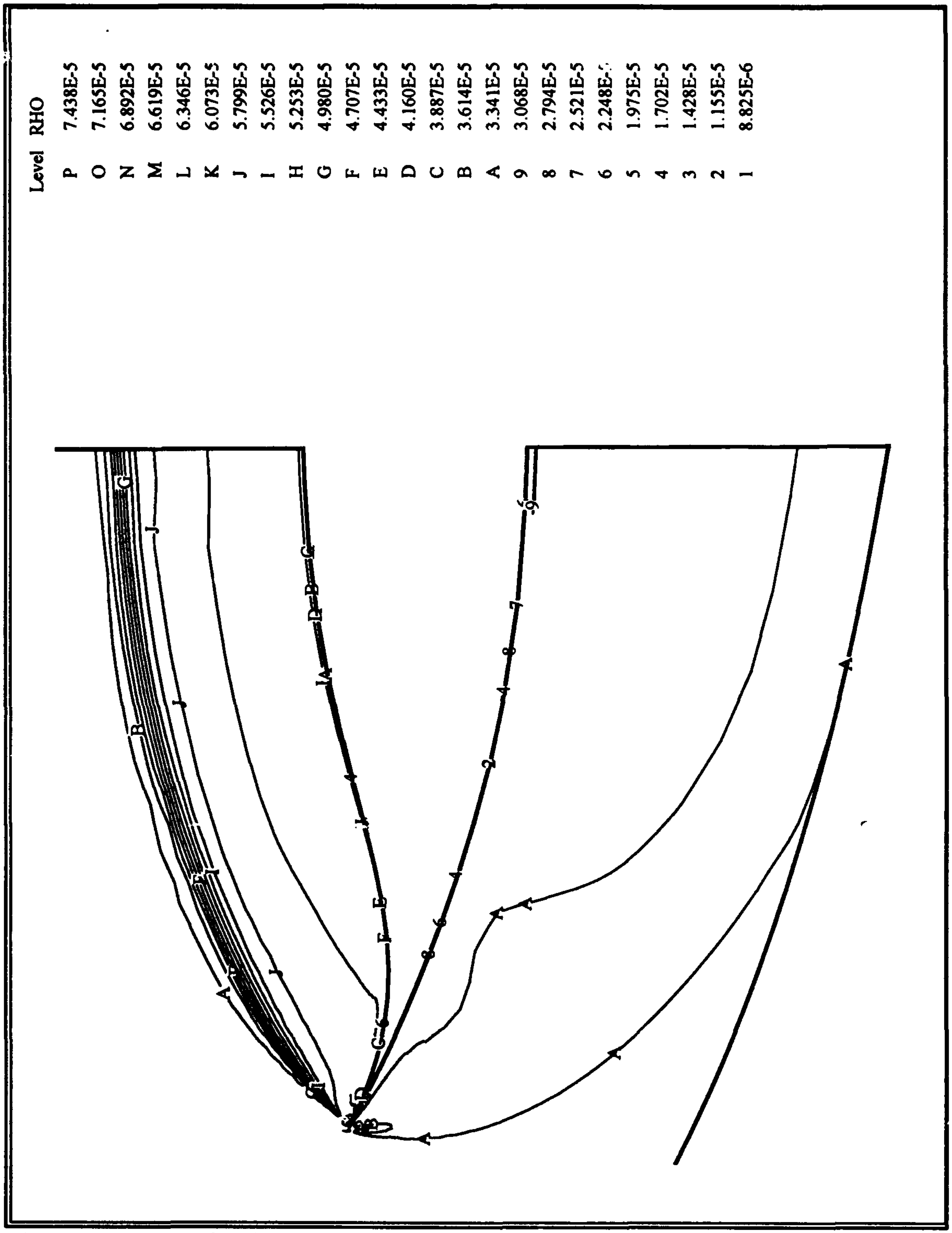

**Figure 4-32**, Density Contours, Viscous Flow, Baseplane

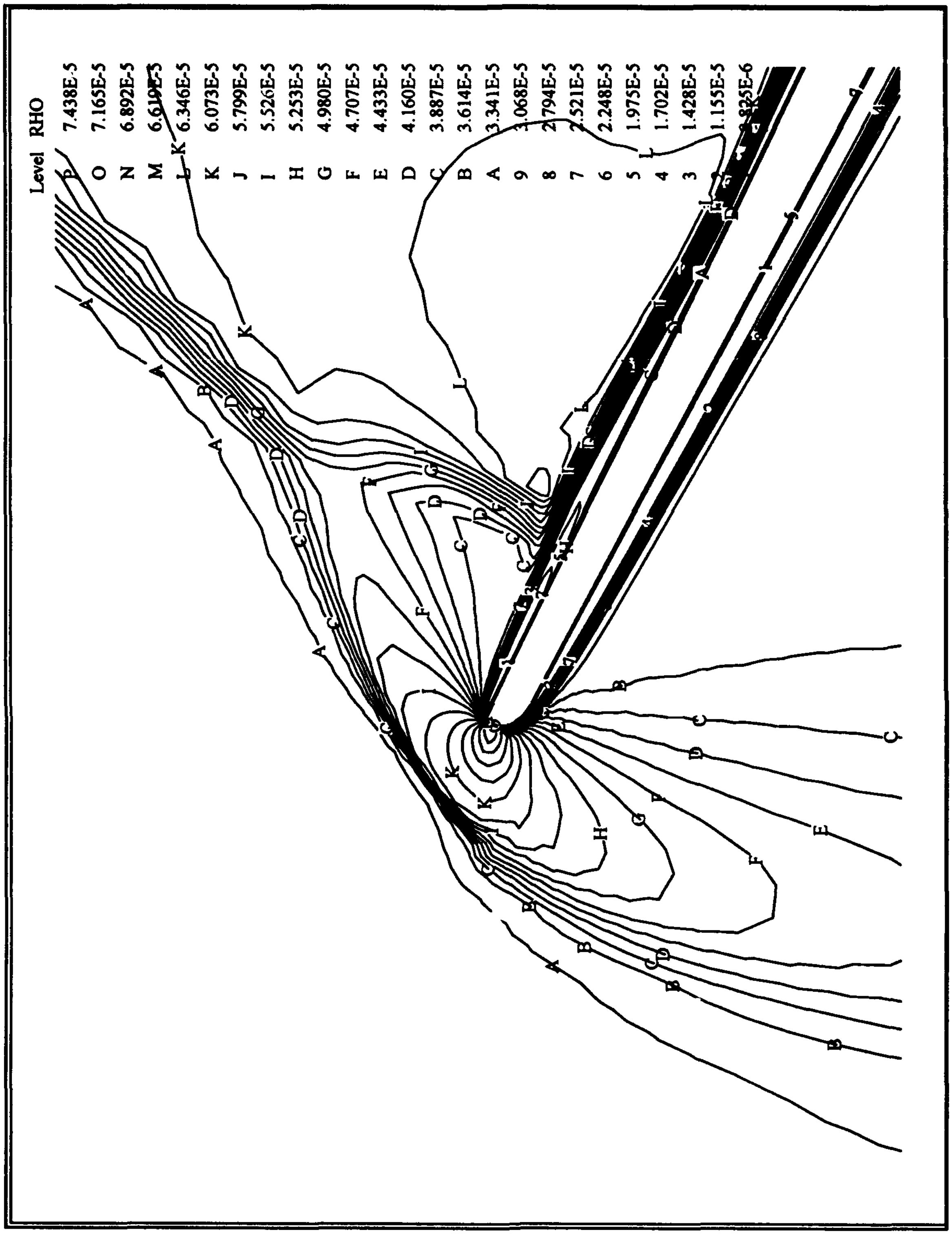

**Figure 4-33**, Density Contours, Viscous Flow, Zoom of Leading Edge

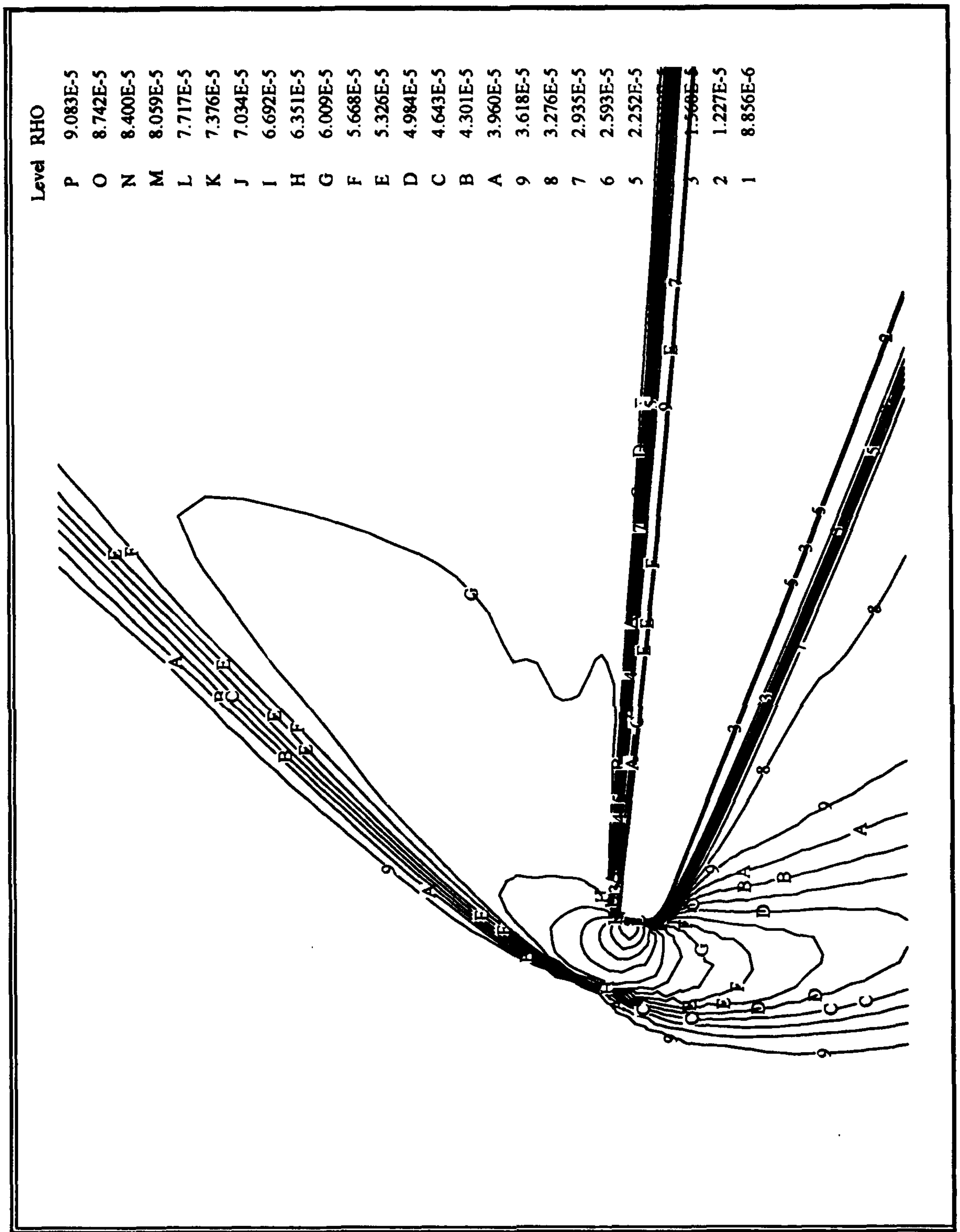

**Figure 4-3**, Density Contours, Viscous Flow, 50th plane from Nose, Zoom of Leading Edge

# V. THEORETICAL AND NUMERICAL ANALYSIS

## 5.1 Boundary Layer Equations

The compressible boundary layer equations are derived directly from the Navier-Stokes equations through order of magnitude analysis. Rasmussen (27) derives the axi-symmetric, steady state, compressible boundary layer equations in some detail, with the final result being:

$$\textit{Continuity:}\qquad \frac{\partial r^{m}\rho u}{\partial x} + \frac{\partial r^{m}\rho v}{\partial y} = 0 \qquad \textbf{(5-1)}$$

Where *m* is 0 for planar two-dimensional flow and 1 for axisymmetric flow  The only difference between the planar equations and the axisymmetric equations is the $r^m$ term in the continuity equation.

$$\textit{X-Momentum:}\qquad \rho u\frac{\partial U}{\partial x} + \rho v\frac{\partial u}{\partial y} = -\frac{dp_e}{dx} + \frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) \qquad \textbf{(5-2)}$$

$$\textit{Y-Momentum:}\qquad \frac{\partial p}{\partial y} = 0 \qquad \textbf{(5-3)}$$

$$\textit{Energy:}\qquad \rho u\frac{\partial h}{\partial x} + \rho v\frac{\partial h}{\partial y} = \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + u\frac{dp_e}{dx} + \mu\left(\frac{\partial u}{\partial y}\right)^2 \qquad \textbf{(5-4)}$$

White (36) derives integral relations for the compressible boundary layer using integral relations similar to those used by Karman. Rasmussen (27) derives a similar relation to that in White.

The integral momentum equation is

$$\frac{d\theta}{dx} + \theta\frac{d}{dx}\left[\ln\left(\rho_e u_e r^m\right)\right] + \left(\theta + \delta^*\right)\frac{d}{dx}\left(\ln u_e\right) = \frac{C_f}{2} \tag{5-5}$$

and the integral enthalpy equation is

$$\frac{d\Phi}{dx} + \Phi\left[\frac{d}{dx}\ln\left(\rho_e u_e r^m\right) + \frac{d}{dx}\ln\left(J_e - J_w\right)\right] = St \tag{5-6}$$

Where $J_w$ and $J_e$ are total enthalpy at the wall and boundary layer edge, respectively, and $St$ is Stanton number. The terms $\delta^*$, displacement thickness, and $\theta$, momentum thickness, are given by

$$\begin{aligned} \delta^* &= \int_0^\delta \left(1 - \frac{\rho u}{\rho_e u_e}\right) dy \\ \theta &= \int_0^\delta \frac{\rho u}{\rho_e u_e}\left(1 - \frac{u}{u_e}\right) dy \end{aligned} \tag{5-7}$$

and $St$ is defined as

$$St = \frac{q_w}{\rho_e u_e \left(J_w - J_e\right)} \tag{5-8}$$

Equations (5-5) and (5-6) can be related through Reynold's analogy. If Prandtl number, Pr, is equal to one, Reynolds analogy states that the left hand sides of equations (5-5) and (5-6) are equal, so that heat transfer is directly related to skin friction. If Pr is not equal to one, Reynold's analogy can be written as (3)

$$St = \frac{C_f}{2\,Pr^{\frac{2}{3}}} \qquad \textbf{(5-9)}$$

For analysis of the stagnation region near the nose, a similarity solution is useful. Such an analysis can be accomplished through the use of the Levy-Lees transformation. Anderson (3) provides a transformation to the Levy-Lees coordinates, $\xi$ and $\eta$:

$$\xi \equiv \int_0^x \rho_e u_e\ \mu_e dx \qquad \textbf{(5-10)}$$

$$\eta \equiv \frac{u_e}{\sqrt{2\xi}} \int_0^v \rho dy \qquad \textbf{(5-11)}$$

Anderson (3) derives the similarity form of the boundary layer momentum and energy equations:

X-Momentum:

$$\frac{\partial}{\partial\eta}(c\ddot{f}) + f\ddot{f} = \frac{2\xi}{u_e}\frac{du_e}{d\xi}\left[f^2 - \frac{\rho_e}{\rho}\right] + 2\xi\left[f\frac{\partial f}{\partial\xi}\ \frac{\partial f}{\partial\xi}\ddot{f}\right] \qquad \textbf{(5-12)}$$

Y-Momentum:

$$\frac{dp}{d\eta} = 0 \qquad \textbf{(5-13)}$$

and Energy:

$$\frac{\partial}{\partial\eta}\left(\frac{c}{Pr}g\right) + fg = 2\xi\left[f\frac{\partial g}{\partial\xi} + \frac{fg}{u_e}\frac{\partial h_e}{\partial\xi} - g\frac{\partial f}{\partial\xi} + \frac{f\rho_e u_e}{\rho h_e}\frac{\partial u_e}{\partial\xi}\right] - c\frac{u_e^2}{h_e}\ddot{f}^2 \qquad \textbf{(5-14)}$$

where

$$g(\xi,\eta) \equiv \frac{h}{h_e} \qquad \textbf{(5-15)}$$

$$\frac{d}{d\eta}\big(f(\xi,\eta)\big) \equiv f = \frac{u}{u_e} \qquad \textbf{(5-16)}$$

and

$$c \equiv \frac{\rho\mu}{\rho_e\mu_e} \qquad \textbf{(5-17)}$$

A computer code (18) which numerically implements the similarity solution, described in equations (5-12) through (5-14), for the stagnation region of hypersonic flow around a blunt ended cylinder was used to compare the flow in the stagnation region to an analytic solution. The following two figures are plots of data produced by the similarity program. Figure 5-1 graphically displays the extreme gradients typical of hypersonic flow. The flow is slowed from 1735 feet/second to 0 feet/second in the space of less than half an inch. The edge velocity is assumed, here, to be equal to that just behind the normal shock.

The similarity solution does not match particularly well with the data taken from the computational solution (Figure 4-31). This is primarily due to poor flow resolution in the stagnation region. A finer computational grid in the stagnation region should

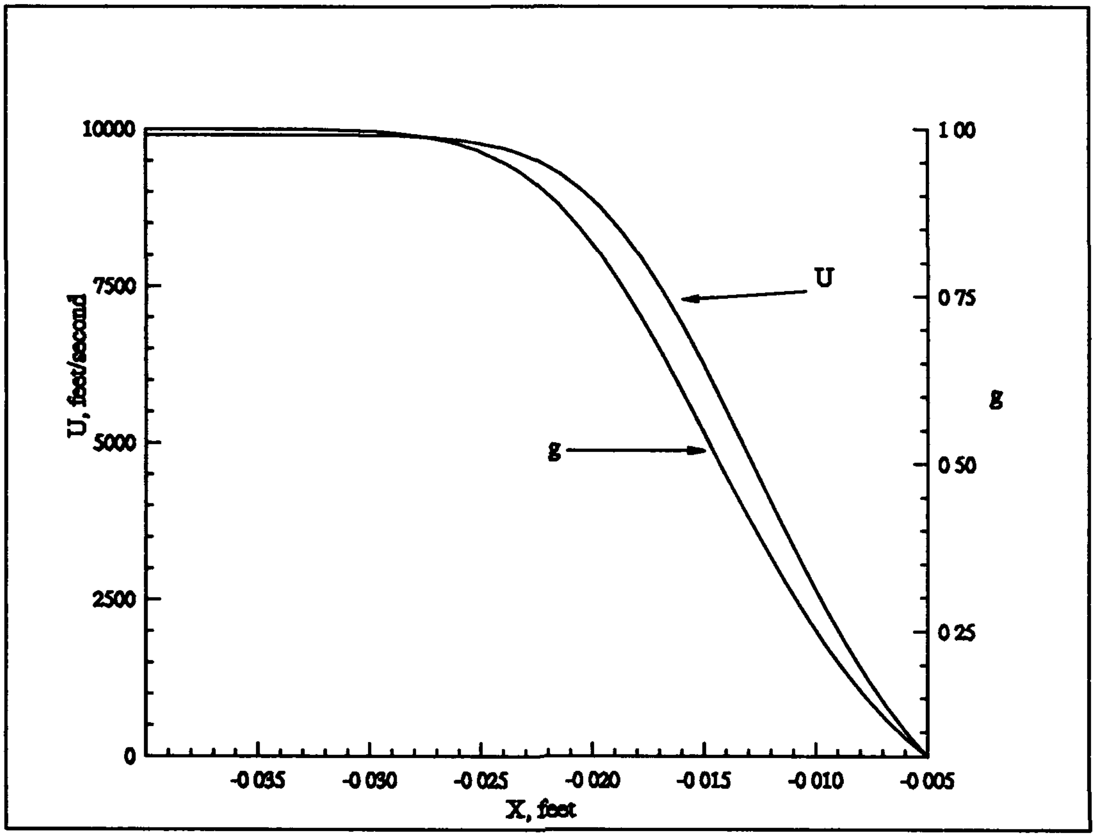

**Figure 5-1,** Stagnation Line Plot of Velocity and g

correct the difficulty. Hayes and Probstein (19) predict that the ratio of shock stand-off distance to nose radius as 0.104, for $M_\infty$=10 and $\gamma$ = 1.4. This matches well with what was observed in the computational solution, but also means that due to the very small shock stand-off distance, the stagnation region will occupy a large portion of the shock layer.

### 5.1.1 Hypersonic Viscous Interaction

Anderson (3) states that the boundary layer thickness in a viscous, hypersonic flow over a flat plate is proportional to Mach number squared divided by the square root of Reynolds number. He gives the relation:

$$\delta \propto \frac{M_e^2}{\sqrt{Re_x}} \qquad \textbf{(5-18)}$$

The waverider, locally, resembles a flat plate, particularly on the freestream surface which is parallel to the freestream. The main difficulty in applying equation (5-18) to the compression surface is its the angle of attack. However, since equation (5-18) is not an equality but a proportionality, it should still hold, though the constant of proportionality will be different than for a flat plate at a zero degree angle of attack.

The rapid boundary layer growth and the resulting thickness of the boundary layer are the basic drivers of hypersonic viscous interaction. Anderson (3) goes on to derive the equation

$$\frac{p_e}{p_\infty} = 1 + a_1\chi \qquad \textbf{(5-19)}$$

for strong viscous interactions, where $a_1$ is a constant. Note that equation (5-19) indicates that the pressure ratio, $p_e/p_\infty$, is dependent only on $\chi$, and that the pressure ratio is linear with $\chi$.

For a weak hypersonic viscous interaction Anderson (3) derives

$$\frac{p_e}{p_\infty} = 1 + b_1\chi \qquad \textbf{(5-20)}$$

where $b_1$ is a constant. Again, the pressure ratio varies linearly with $\chi$. Anderson (3) provides numerical values for the constants $a_1$ and $b_1$ for hypersonic flow over both an adiabatic wall and a cold wall flat plate. Equations (5-19) and (5-20) become:

$$\frac{p}{p_\infty} = 1 + 0.5\chi \qquad (\textit{strong interaction})$$

$$\frac{p}{p_\infty} = 1 + 0.078\chi \qquad (\textit{weak interaction}) \tag{5-21}$$

for the cold wall case. With the viscous interaction parameter $\chi > 3.0$, strong interaction is generally taken to occur, and weak interaction for $\chi < 3.0$ (3). Note that the strong interaction case varies much more rapidly with $\chi$ than does the weak interaction case because the constant multiplying $\chi$ is an order of magnitude larger for the strong interaction case than for the weak interaction case.

The definition of $\chi$ in the above equations is:

$$\chi \equiv \sqrt{C}\,\frac{M_\infty^3}{\sqrt{Re}} \tag{5-22}$$

where $C$ is given by

$$C \equiv \frac{\rho_w \mu_w}{\rho_e \mu_e} \tag{5-23}$$

### 5.1.2 Application of the Boundary Layer Equations

The above equations were applied to the waverider flying at Mach 10 and $\alpha$ = 0 at an altitude of 100,000 feet. Figure 5-2 shows the estimated variation of $\chi$ with distance from the waverider's nose. For purposes of calculating $\chi$ in Figure 5-2 the following assumptions: $u_e = .9u_{fs}$ = 8910 feet/second, $T_e$ = 1469 °Rankine, and $C$ = 1.25 were made. Sutherland's formula was used to calculate $\mu_\infty$ and $Re$ was calculated using the value of the variables at the boundary layer's edge.

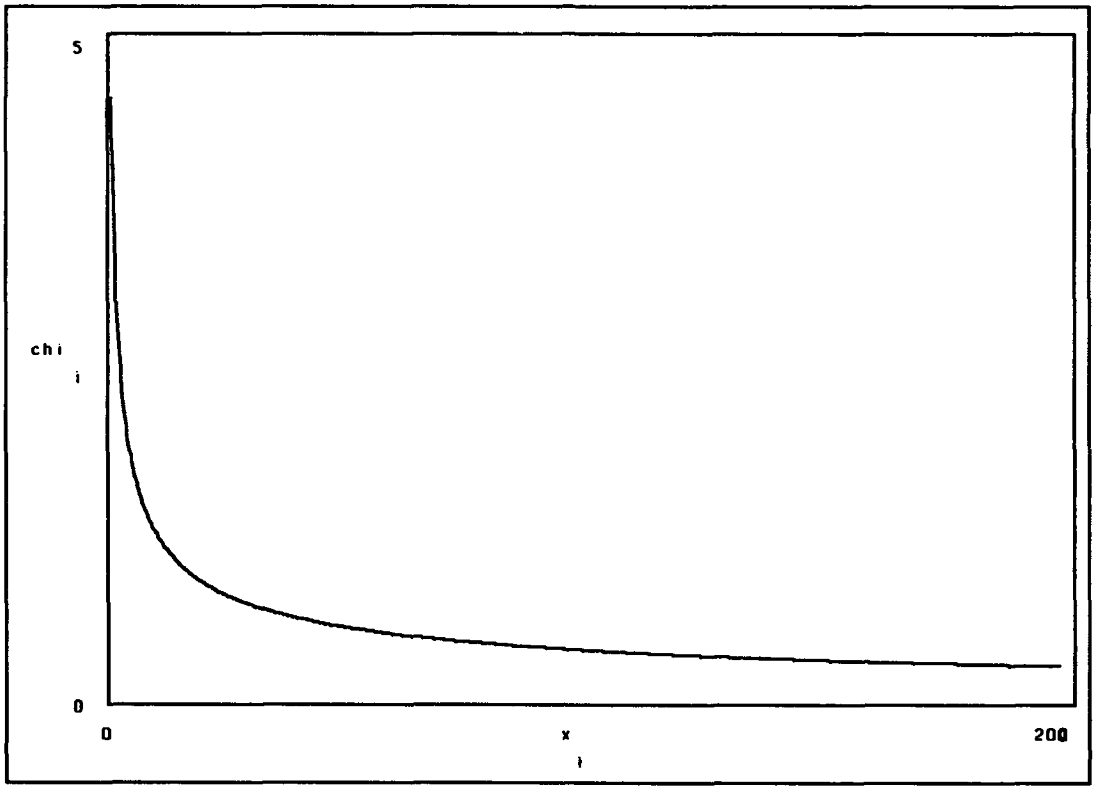

**Figure 5-2,** Chi vs X

Applying $\chi$ as shown in Figure 5-2 to equation (5-21), the local pressure ratio is calculated. The pressure ratio distribution over *x*, calculated both from the computational data and from the viscous interaction equations given above, is shown in Figure 5-3 (compression surface) and in Figure 5-4 (freestream surface).

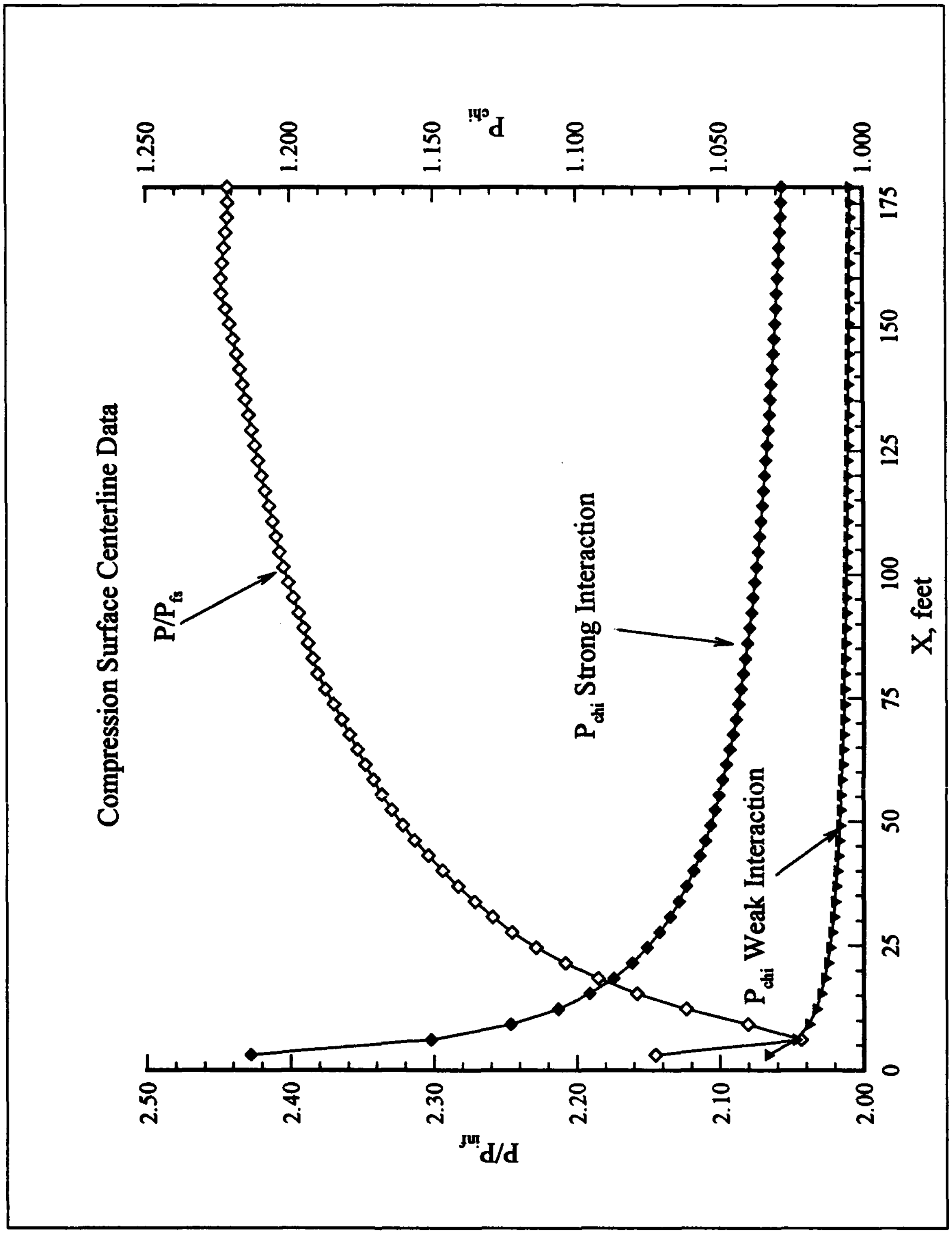

**Figure 5-3**, Compression Surface Centerline Pressure Ratio

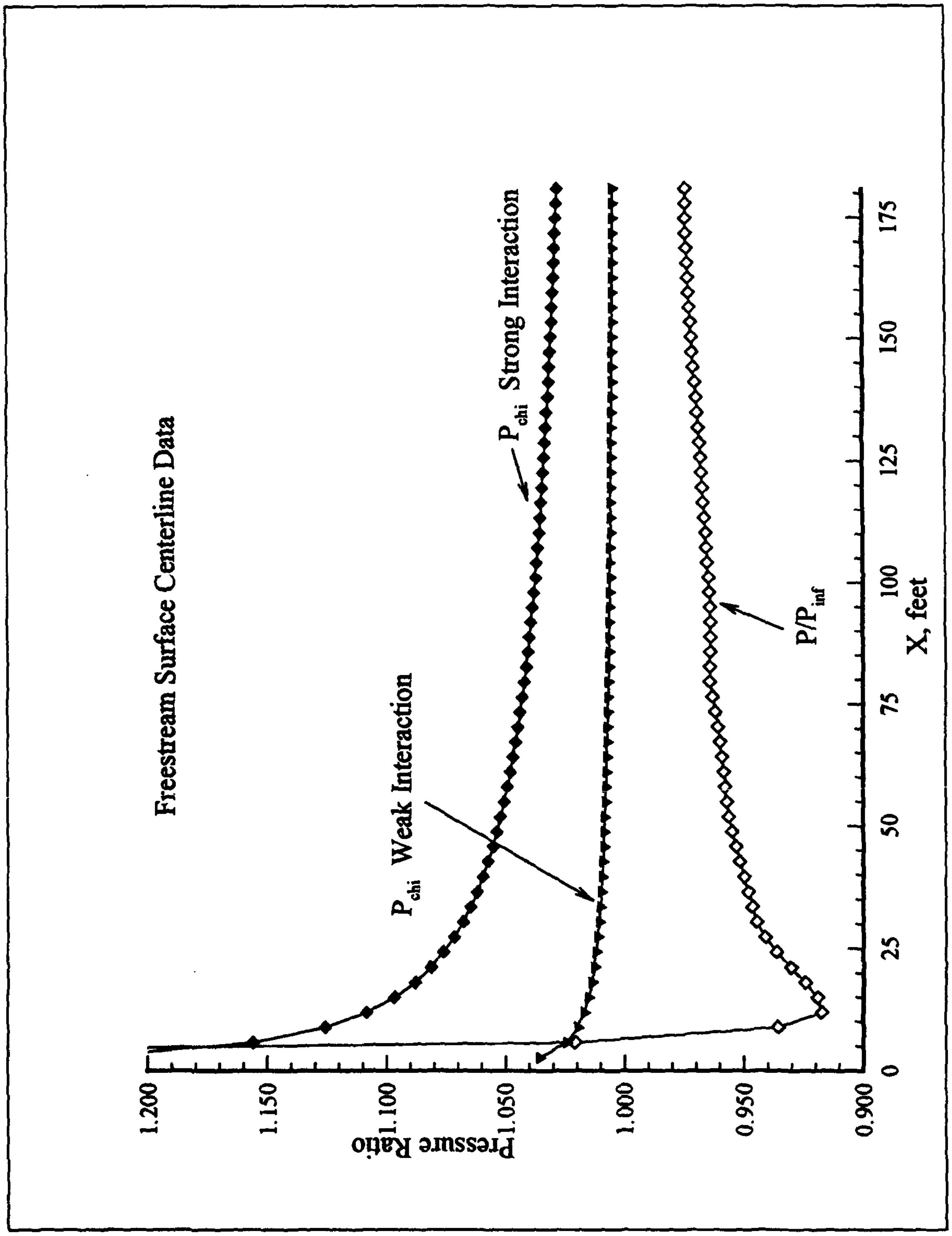

**Figure 5-4**, Freestream Centerline Pressure Ratio,

Appropriate flow variables, particularly $\rho_w$, were extracted from the computational solution for use in the calculation of $\chi$.

As can be seen in both figures, the match is not particularly good. The initial trend of a steep decrease in pressure ratio is correct, though the values don't match. The numerical results are initially greater than the theoretic prediction. The steep pressure rise behind the normal shock accounts for the displacement of the computational pressure ratio above the predicted ratio. The sudden turn and then rise in the computationally calculated pressure ratio is due to the flow expanding past freestream conditions around the blunt nose, and then trying to recover to the freestream pressure, as described earlier, in Chapter 4.

### 5.2 Skin Friction and Heat Transfer

Incompressible flow over a flat plate produces the following skin friction and heat transfer coefficients (36) derived from similarity solutions:

$$c_f = \frac{0.664}{\sqrt{Re_x}} \qquad \textbf{(5-24)}$$

and

$$St = \frac{0.332}{\sqrt{Re_x}\, Pr^{\frac{2}{3}}} \qquad \textbf{(5-25)}$$

The parameters Re and Pr are based on the flow properties at the edge of the boundary layer.

The reference temperature method uses the exact same form for the skin friction and heat transfer coefficients as is found in equations (5-24) and (5-25), except that Re and Pr are now calculated using a reference temperature, T$^*$, given by Anderson (3) as:

$$\frac{T^*}{T_e} = 1 + 0.032M_e^2 + 0.58\left(\frac{T_w}{T_e} - 1\right) \qquad \textbf{(5-26)}$$

Thus, Re and Pr can be written as:

$$Re_x^* = \frac{\rho^* u_e x}{\mu^*} \qquad Pr^* = \frac{\mu^* c_p^*}{k^*} \qquad \textbf{(5-27)}$$

The values in equation (5-27) are then substituted directly into equations (5-24) and (5-25). The reference values used in equation (5-27) were taken from the just behind the shock at about the midpoint of the waverider, to ensure that the reference values used were indeed representative of the conditions at the edge of the boundary layer.

The plot of skin friction coefficient versus x (feet) in Figure 5-5 shows results very similar to those returned by the computational solution, shown in Figure 5-6. The agreement between the two solutions is good. Both figures show precipitous drop in both heat transfer and skin friction as distance from the nose increases. Both plots asymptotically approach a value of 0.00004 for skin friction coefficient and .00008 for Stanton number. Agreement with Reynold's analogy in the numerical solution is also good, Stanton number being a little less than twice the value of the skin friction coefficient.

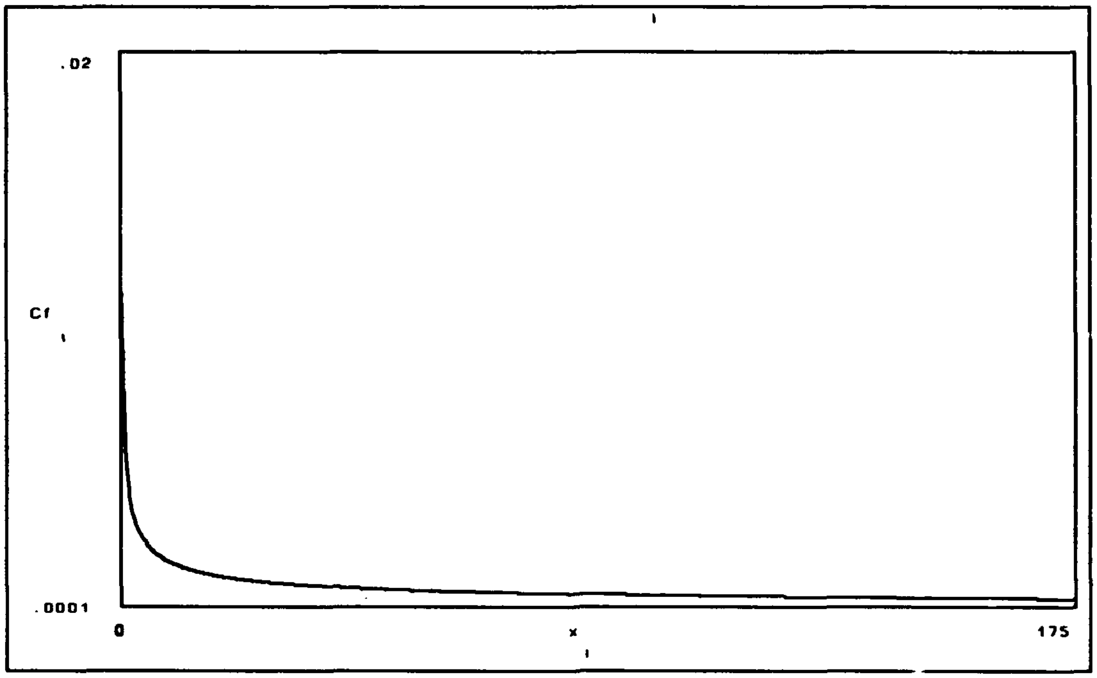

Figure 5-5, $c_f$ versus x

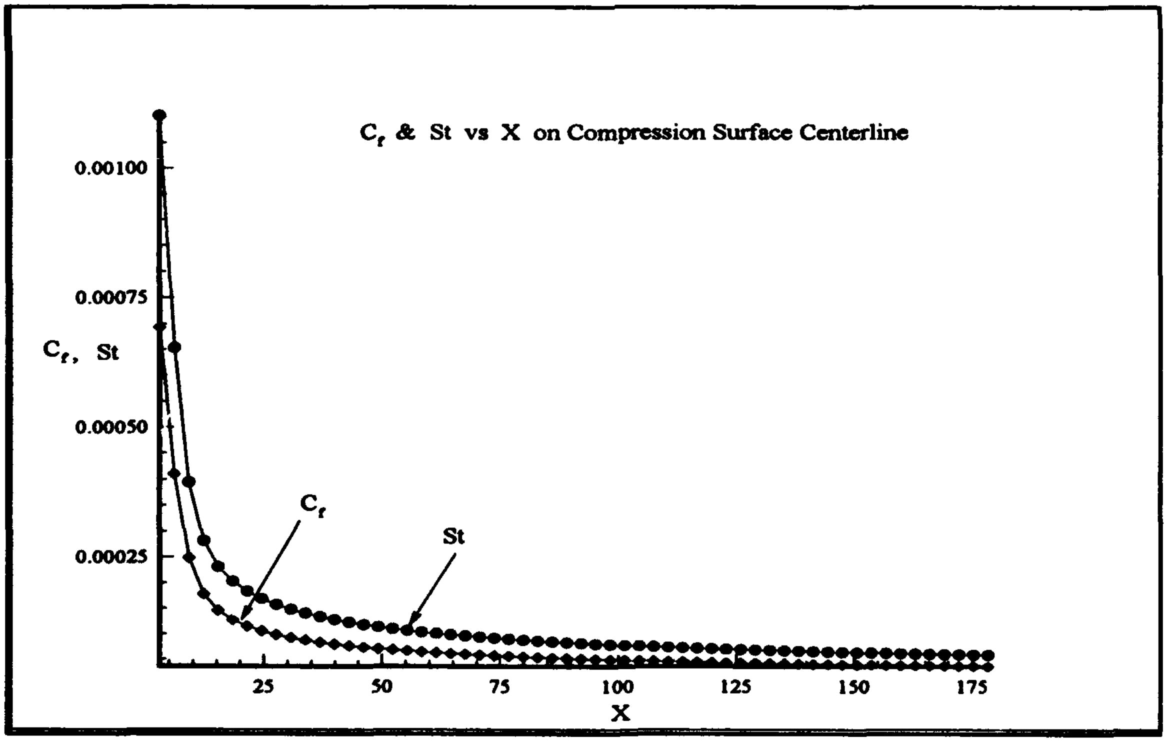

Figure 5-6, St & $c_f$, Compression Surface Centerline

## 5.3 Lift and Drag

The lift and drag forces are calculated by integration of the pressure and shear stresses over the body. White (36) gives the control volume equation:

$$F = \iint_S \frac{dF_s}{dA} + \iiint_{Vol} \frac{dF_b}{dm} \rho \, dVol \qquad \textbf{(5-28)}$$

where $\boldsymbol{F_s}$ is the surface force and $\boldsymbol{F_b}$ is the body force. The total force, $\boldsymbol{F}$, is then broken into lift and drag components by simply taking lift as the component of $\boldsymbol{F}$ in the negative Y direction (see Figure 1-2) and drag as the component in the direction of the freestream flow. A program, written by Gaitonde, which analyzes the data produced by the CFD solver further breaks the lift and drag components into viscous and pressure components. Table 5-1 lists the forces for the viscous and inviscid cases normalized to the force in the X direction.

**Table 5-1, Force Data**

| Parameter | Viscous Flow Value | Inviscid Flow Value |
|---|---|---|
| Total Lift, lbs | 223198 | 195805 |
| Total Drag, lbs | 38853 | 23999 |
| Pressure Lift, lbs | 223639 | 195805 |
| Pressure Drag, lbs | 27708 | 23999 |
| Viscous Lift, lbs | -441 | 0 |
| Viscous Drag, lbs | 11145 | 0 |
| $P_{max}$, psf | 149.938 | 142.593 |
| $P_{\infty}$, psf | 23.276 | 23.276 |
| $Re_L$ | 193,043,000 | |
| L/D | 5.745 | 8.159 |

The flight conditions outlined in Table 4-1 were used in the calculation of both the viscous and inviscid cases.

As can be seen, the viscous terms contribute significantly to the drag on the vehicle, almost 30% of the drag being due to viscosity. The viscous effects increase lift by 12% as compared to the inviscid case. The increase in lift is due to the slightly greater pressure on the compression surface, caused by the hypersonic-boundary layer interaction present in the viscous flow solution.

# VI. CONCLUSIONS AND RECOMMENDATIONS

## 6.1 Conclusions

Mach 10 Navier-Stokes flow over a conically derived hypersonic waverider was computationally solved on the OSC CRAY Y-MP using WL/FIM's implicit flux splitting code. The code performed well, its finite volume formulation enhancing the practical stability, despite some very skewed cell geometries.

Returning to the objectives stated in chapter I, the following conclusions were reached:

Blunting the nose and leading edges had little effect on results such as lift to drag ratio, though it did significantly effect the flow structure, particularly near the nose of the vehicle.

Generating the three dimensional grid was a major effort, though made considerably simpler by the GRIDGEN package. The use of the O-grid allowed for good resolution near the waverider's leading edges, allowing the capturing of flow expansion and shock standoff.

The three dimensional implicit flux splitting algorithm performed extremely well. The finite volume approach the code used made for an extremely robust algorithm, able to handle regions with highly skewed cells, and cells with very small volumes. The code produced results predicted by hypersonic theory, with deviations attributable to numerical error in either the waverider model or in the numerical solution procedure. The implicit version of the code required fewer iterations to reach convergence than did the explicit version. The explicit version, however, required less computation time per iteration and less memory overhead. The tradeoffs between the versions of the code favored the implicit version. While each iteration for the implicit version required approximately twice as much computer time, the implicit version required approximately one quarter the number of iterations. If memory becomes a limiting factor, however, the explicit version of the code has a definite advantage.

The blunted nose and leading edges led to a small, but strong, normal shock slightly displaced from the waverider. The pressure rise through the normal shock, and the expansion around the shoulder of the waverider caused the flow to differ significantly from the results predicted by the viscous interaction equations given by Anderson (3), making the comparison of little value. The structure of the shock waves was simple and held only one minor surprise, the $\lambda$ shaped structure at the leading edge on the baseplane, formed by an expansion around the leading edge meeting the high pressure on the compression surface. Elsewhere, the shock was detached from the waverider by a small distance.

Comparison of the stagnation region near the nose to a similarity solution for flow over a similar geometry yielded good results. The trends matched well, and the values computed were close to those predicted. Differences can easily be attributed to differences in the geometry, and highlight the need for a finer grid in the stagnation region.

The computed *L/D* ratio compared favorably with results for similar vehicles. Comparing the result of 5.75 obtained in this research to a plot of *L/D* such as that in

Anderson (5), this result can be seen to be quite near the predicted value, and the results obtained by others.

### 6.2 Recommendations

This thesis, along with Stecklein's (28), will provide an excellent basis from which to continue waverider research. The following recommendations are the result of careful consideration of the research and conclusions reached in the course of this thesis.

The use of a taut cubic spline, or radius method to blunt the nose and leading edge is recommended. Such a method will provide a smoother model geometry at the nose and leading edges than that used here. Refining the computational grid is necessary, particularly in the stagnation region near the waverider's nose. Some particular goals to pursue include reducing the number of cells in the grid without losing the flow resolution, and reducing the cell skewness, particularly near the leading edges.

This research neglected the real gas effects, such as dissociation, chemical reaction, and ionization. Murthy (23), and or Anderson (3), describe these effects the influence they have on the flow, but the most noticeable effects are a reduction in the boundary layer thickness and in aerodynamic heating. Turbulence should be accounted for in further studies. The Reynolds number indicates that turbulence could play a major role in the flow over the waverider.

Flight at off-design conditions should be examined. Waveriders are very sensitive to off-design conditions, and pay heavy penalties in terms of efficiency when flying at other than their design point. Particularly, waverider performance over a range of Mach numbers from 0 to somewhat above the design Mach number should be investigated, as should performance for a range of angles of attack and yaw. Waverider performance over a range of altitudes, perhaps a reentry trajectory, should be investigated.

# APPENDIX A: DERIVATION OF THE WAVERIDER EQUATIONS

The waverider equations are well known, as is their derivation. However, most derivations leave large steps to the reader's imagination and frustration. Therefor, the equations are derived here in some detail. The derivations came primarily from Rasmussen (25 and 26) and Stecklein (28).

The conically derived waverider equations come from the Taylor-MacColl equation and it's hypersonic small disturbance theory form and results.

$$\frac{\gamma-1}{2}\left[V_\infty^2 - V_r^2 - \left(\frac{dV_r}{d\theta}\right)^2\right] \cdot \left[2v_r + \frac{dV_r}{d\theta}\cot\theta + \frac{d^2V_r}{d\theta^2}\right] - \frac{dV_r}{d\theta}\left[V_r\frac{dV_r}{d\theta} + \frac{dV_r}{d\theta}\left(\frac{d^2V_r}{d\theta^2}\right)\right] = 0 \tag{A-1}$$

Equation (A-1) is the Taylor-MacColl equation, which produces the HSDT form:

$$\frac{\partial^2 V_r}{\partial\theta^2} + \cot\theta\frac{\partial V_r}{\partial\theta} + 2V_r = 0 \tag{A-2}$$

Equation (A-2) has analytic solutions for u and v ($V_r$, and $V_\theta$ respectively), which, using 2nd order accurate Taylor series expansion, are:

$$u \approx V_\infty\left[1 - \frac{\delta^2}{2}\left(\frac{\theta^2}{\delta^2} + \ln\frac{\beta^2}{\theta^2}\right)\right]$$
$$v \approx -V_\infty\theta\left(1 - \frac{\delta^2}{\theta^2}\right) \tag{A-3}$$

The streamlines between the shock and the cone (i.e. within the shock layer) are given by

$$V \times ds = 0 \tag{A-4}$$

where *ds* is differential length along a streamline, and *V* is the velocity vector. The equation says that the velocity vector is parallel to the streamline.

Expanding (A-4) results in only one component, that in the $\phi$ direction:

$$\det\begin{bmatrix} \hat{e}_r & \hat{e}_\theta & \hat{e}_\phi \\ u & v & 0 \\ dr & rd\theta & 0 \end{bmatrix} = (u\,rd\theta - v\,dr)\hat{e}_\phi = 0 \tag{A-5}$$

Rearranging (A-5) by getting rid of the unit vector and separating the variables you get:

$$\frac{dr}{r} = \frac{u}{v}d\theta \tag{A-6}$$

Further assuming that $u \approx V_\infty$ and is furthermore constant within the shock layer, (A-6) can be integrated (after substituting for v from (A-3)) from the shock to a point of interest:

$$\int_{r_s}^{r}\frac{dr}{r} = -\int_{\beta}^{\theta}\frac{\theta d\theta}{\theta^2-\delta^2} \qquad \textbf{(A-7)}$$

(A-7) can be solved to become:

$$lnr - lnr_s = \frac{1}{2}\ln|\beta^2-\delta^2| - \frac{1}{2}\ln|\theta^2-\delta^2| \qquad \textbf{(A-8)}$$

which can be rearranged by raising both sides to the *e* to:

$$r_s = r\sqrt{\frac{\theta^2-\delta^2}{\beta^2-\delta^2}} \qquad \textbf{(A-9)}$$

Now, the distance from the centerline of the cone to any given point is $rsin\theta$ or, for small angles, $r\theta$. The (arbitrary) cylindrical freestream surface can be expressed by:

$$r\sin\theta = f(\phi) \qquad \textbf{(A-10)}$$

where $f(\phi)$ is arbitrary. Applying the small angle approximation, the freestream surface which intersects the shock at $r_s$ is given by:

$$r\theta = r_s(\phi)\beta \qquad \textbf{(A-11)}$$

In the base plane we non-dimensionalize by $l\delta$, the radius of the generating cone in the base plane. This gives $R_{\infty b} = \theta_{\infty b}/\delta$. If (A-11) is evaluated in the base plane it yields:

$$r\theta = l\theta_{\infty b}(\phi) = r_s(\phi)\beta \qquad \textbf{(A-12)}$$

Defining $\sigma$ as $\beta/\delta$ (A-12) can be rearranged to get:

$$r_s(\phi) = \frac{l}{\sigma} R_{\infty b}(\phi) \qquad \textbf{(A-13)}$$

Now, $R_{cb}(\phi)=l\theta_{cb}(\phi)/l\delta=\theta_{cb}(\phi)/\delta$, or $\theta_{cb}=\delta R_{cb}(\phi)$. Evaluating (A-9) in the baseplane, with (A-13) substituted in for $r_s(\phi)$, yields:

$$\frac{R_{\infty b}(\phi)}{\sigma} = \sqrt{\frac{\theta_{cb}^2 - \delta^2}{\beta^2 - \delta^2}} \qquad \textbf{(A-14)}$$

Squaring both sides, rearranging slightly, and substituting in for $\theta_{cb}$:

$$\delta^2 R_{cb}^2 - \delta^2 = R_{\infty b}^2(\phi)\frac{\beta^2 - \delta^2}{\sigma^2} \qquad \textbf{(A-15)}$$

A little algebra gives the final result:

$$(R_{cb}(\phi))^2 = 1 + \left(\frac{\sigma^2 - 1}{\sigma^2}\right)(R_{\infty b}(\phi))^2 \qquad \textbf{(A-16)}$$

Note that $R_{\infty b}$ is an arbitrary function, usually (and in this thesis) taken from the transformation of the polynomial:

$$X = R_0 + AY^2 + BY^4 + CY^6 \qquad \textbf{(A-17)}$$

# APPENDIX B: DIVIDED DIFFERENCES

| | $[\,]P_i$ | $[\,,\,]P_i$ | $[\,,\,,\,]P_i$ | $[\,,\,,\,,]P_i$ |
|---|---|---|---|---|
| $\tau_i$ | $g(\tau_i)$ | | | |
| | | $s_i$ | | |
| $\tau_i$ | $g(\tau_i)$ | | $([\tau_i,\tau_{i+1}]g-s_i)/\Delta\tau_i$ | |
| | | $[\tau_i,\tau_{i+1}]g$ | | $(s_{i+1}+s_i-s[\tau_i,\tau_{i+1}]g)/(\Delta\tau_i)^2$ |
| $\tau_{i+1}$ | $g(\tau_{i+1})$ | | $(s_{i+1}-[\tau_i,\tau_{i+1}]g)/\Delta\tau_i$ | |
| | | $s_{i+1}$ | | |
| $\tau_{i+1}$ | $g(\tau_{i+1})$ | | | |

**Table B-1, Divided Difference Table**

The square brackets, [ ], indicate a divided difference of order *i-1*. The first divided difference is given in Eqn (B-2) below.

$$f[x_0,x_1] = \frac{f_1 - f_0}{x_1 - x_0} \qquad \textbf{(B-2)}$$

The second divided difference incorporates the first:

$$f[x_0,x_1,x_2] = \frac{f[x_1,x_2] - f[x_0,x_1]}{x_2 - x_0} \qquad \textbf{(B-2)}$$

Similarly for the third divided difference.

The values given in Table B-1 are used to derive the coefficients in Eqn (2-11).

## APPENDIX C: PROGRAM WAVERIDER

```
c
c  5 Sep 92:
c      This attempt will now proceed as follows:
c  Both freestream and compression surfaces will be extended as y=y(z).
c  The slopes of each will be clamped at -3 for the compression and +3
c  for the freestream surfaces, at the boundary point x5,y5.  However,
c  the boundary point value will not actually be written to the file,
c  thus blunting the sharp corner that would otherwise result.
c      The other major change is that the first plane, the i=1 plane,
c  will not be printed to the output file.  Instead, I3G-VIRGO will be
c  used to generate the cubic spline on the nose.  This will hopefully
c  eliminate singularity problems near the nose.
c      Note also that the slopes may be input, as can the separation
c  distance and the number of points to use in the spline.  Good results
c  were obtained using 3cm in y, 1cm in z, slopes of +-4, and 15 points.
c
c  23 Sep 92:
c      The program currently uses linear interpolation to build the
c  nose of the waverider.
c

      PROGRAM WVRIDR

C      PRODUCES MULTIPLE 2-D CROSSPLANE SURFACES OF A
C      PARABOLIC HYPERSONIC WAVERIDER

      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER MCAP,NCAP,INCR,BNDS
      REAL*8 MACH,G,D,PHIL,l,SIGMA,Ro,Ao,XSIG,YSIG,PI,
     *       lw,rz,RAD,ZovL(90),X(90,91),Y(90,91),Z(90,91)

      CHARACTER*21 OUTPUT
      COMMON MCAP,NCAP,PI,INCR,BNDS,PHIL

C      SET CONSTANTS

      PI = 4.d0*DATAN(1.d0)
      RAD = PI/180.

C      CONSTANTS WHICH EFFECT THE GEOMETRY OF THE WAVERIDER
      MACH = 10.0
      D = 5.5 * RAD
      PHIL = 50.0 * RAD
      l = 88.41675
      TYPE = 1

C     BASIC CALCULATIONS NEEDED FOR WAVERIDER CROSSPLANE
C     DEVELOPMENT

      G = 1.4
      MCAP = 45
      NCAP = 90

      INCR = (2*MCAP) + 1
      BNDS = 1

      OUTPUT = ’3dsurf’

      SIGMA = ((G+1.0)/2.0 +1.0/((MACH*D)**2))**0.5

      XSIG = SIGMA*COS(PHIL)
      YSIG = SIGMA*SIN(PHIL)

C          VALUE OF TYPE DEFINES TYPE OF PARABOLIC WAVERIDER.
DEFAULT
C     IS SET TO TANGENT PARABOLIC

      IF (TYPE.EQ.0) THEN
        Ro = XSIG/2.0
        Ao = Ro/(YSIG)**2
       ELSE
        Ro = 0.75*XSIG
        Ao = 0.25*XSIG/(YSIG)**2
      END IF

      lw = l*(1.0 - Ro/SIGMA)
      rz = l*(Ro/SIGMA)

C     GET INPUT DATA
C     CALL INPUT(MACH,D,PHIL,l,TYPE,MCAP,NCAP,OUTPUT)

C     COMPUTE WAVERIDER GEOMETRY
      CALL WAVEBODY(Ro,Ao,l,D,SIGMA,rz,Rin,Rcb,ZovL,
     *              X,Y,Z)
```

```
C     CALL OUTDAT TO WRITE TO A DATA FILE
      CALL OUTDAT(Rin,Rcb,lw,l,MACH,D,ZovL,
     *           X,Y,Z)

      END
C     END OF MAIN PROGRAM
```

```
************************************************************************
      SUBROUTINE WAVEBODY(Ro,A,l,D,SIGMA,rz,Rin,Rcb,
     *                    ZovL,X,Y,Z)

C     WAVEBODY DEFINES THE X,Y,& Z COORDINATES OF CROSSPLANES
C     OF THE WAVERIDER CONFIGURATION

      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER I,J,K,INCR,M,N,BNDS
      REAL*8 PHI(90),PHIZ(90),Rin,rz,rs(90),ZovL(90),
     *     Rcb,Xin(90,60),Yin(90,60),Zin(90,60),
     *     Xcb(90,60),Ycb(90,60),Zcb(90,60),A,Ro,l,
     *     SIGMA,test,X(90,91),Y(90,91),
     *     Z(90,91),D,DELTA(90),A1,PLANE(90),A0,PROG

      COMMON MCAP,NCAP,PI,INCR,BNDS,PHIL

C     SET INITIAL VALUE OF PHI TO COINCIDE WITH THE
C     NUMBER OF CROSSPLANES: (PHIL/RAD)/NCAP

      A0 = .001
      PHI(1) = 0.0*PI/180.
      PROG = PHIL-PHI(1)
       CALL GEOM(PROG,A0,NCAP-1,PLANE)
        DO N = 2,NCAP
         PHI(N) = PHI(N-1) + PLANE(NCAP+1-N)
        END DO
      PHI(NCAP) = PHIL

        scale = l*D

      DO 10 I = 1,NCAP

C     CORRECT FOR ERRORS IN MACHINE ZERO

      test = ((COS(PHI(I)))**2-4.0*Ro
     *          *A*(SIN(PHI(I)))**2)

      IF(test.LE.0.0)THEN
        test = 0.0
      END IF

      rs(I) = (l/SIGMA)*(2.*Ro)/(COS(PHI(I))+(test)**0.5)
      ZovL(I) = (rs(I)-rz)/(l*(1.0-Ro/SIGMA))

C     A1 DETERMINES THE INITIAL PACKING INCREMENT OF BODY POINTS
C      FROM THE LEADING EDGE OUTWARD.

        A1 = .0005

C      COMPUTE THE PHI INCREMENTS FOR BODY POINT GENERATION
        STEP = PHI(I)
      CALL GEOM (STEP,A1,MCAP,DELTA)

         PHIZ(1) = 0.0
      DO 15 J = 2,MCAP+1

   15     PHIZ(J) = PHIZ(J-1) + DELTA(MCAP+2-J)
         PHIZ(MCAP+1) = PHI(I)

       DO 20 J = 1,MCAP+1
          K = (I-1)*(MCAP+1)+J
          if (k .eq. 1) then
            Rin1 = ((2.* Ro)/(COS(PHIZ(J))+((COS(PHIZ(J)))**2
     *         - 4.*Ro*A*(SIN(PHIZ(J)))**2)**0.5))
            end if

      Rin = ((2.* Ro)/(COS(PHIZ(J))+((COS(PHIZ(J)))**2
     *         - 4.*Ro*A*(SIN(PHIZ(J)))**2)**0.5))

      Rcb = (((ZovL(I)*(1.0-Ro/SIGMA)+Ro/SIGMA)**2 +
     *         (SIGMA**2 - 1.0)*Rin**2/SIGMA**2)**0.5)

C      CALCULATE THE X,Y,& Z VALUES FOR EACH CROSSPLANE.
C       Z IS A CONSTANT FOR EACH CROSSPLANE

      Xin(I,J) = Rin*COS(PHIZ(J))
      Yin(I,J) = Rin*SIN(PHIZ(J))
      Zin(I,J) = rs(I)

      Xcb(I,J) = Rcb*COS(PHIZ(J))
      Ycb(I,J) = Rcb*SIN(PHIZ(J))
      Zcb(I,J) = rs(I)

   20 CONTINUE
```

```
 10 CONTINUE

C   COMBINE THE X, Y, & Z VALUES OF THE FREESTREAM AND
C   COMPRESSION SURFACES INTO ONE ARRAY

   DO 30 I = 1, NCAP
    DO 40 J = 1,MCAP

     Y(I,J) = Xin(I,J)*scale - Rin1
     Z(I,J) = Yin(I,J)*scale
     X(I,J) = Zin(I,J) - (l*Ro/SIGMA)

 40  CONTINUE
 30 CONTINUE

   DO 50 M = 1,NCAP
    DO 60 N = MCAP+1,INCR

     Y(M,N) = Xcb(M,(INCR+1)-N)*scale - Rin1
     Z(M,N) = Ycb(M,(INCR+1)-N)*scale
     X(M,N) = Zcb(M,(INCR+1)-N) - (l*Ro/SIGMA)
 60  CONTINUE

 50 CONTINUE

   RETURN
   END
```

```
************************************************************************
      SUBROUTINE OUTDAT(Rin,Rcb,lw,l,MACH,D,ZovL,X,Y,Z)
C  OUTPUT SENDS RESULTS TO AN OUTPUT FILE
c  This subroutine also adds the spline fit required for a
c  viscous waverider.
c
      INTEGER I,J,INCR,BNDS,mm,dim, ibcbeg, ibc1, nose
      REAL*8 X(90,91),Y(90,91),Z(90,91),lw,MACH,D,Rin,Rcb,ZovL(90)
      REAL*8 h,x1,x2,x3,x4,y1,y2,y3,y4,slope,displ
      REAL*8 xsp(20), ysp(20), tau(2), c(4,2), sep, xsep
      REAL*8 xnose(20),ynose(40),znose(20),yspcb(20)

      COMMON MCAP,NCAP,PI,INCR,BNDS,PHIL

      OPEN(UNIT=12,file='visc.dat',form='formatted')
      OPEN(UNIT=14,file='curve.dat',form='formatted')
c       write(12,*) 'NETWORK=WAVERIDER',mcap*2+19,ncap, '  NEW'

      Print*, 'Enter distance (meters) to separate surfaces'
      Read*, sep

      xsep = 0.5 * sep
       y1=y(1,1)
c
c
c Split the two surfaces by 1 cm. and put the waverider nose at Y=0
c
c
      Do 10 j=1,ncap
      Do 20 i=mcap+1,mcap*2+1
            y(j,i)=y(j,i)+sep
20       continue
10      continue
      do 15 j=1,ncap
        do 25 i=1,mcap*2+1
          y(j,i)=y(j,i)-y1-xsep
25       continue
15      continue

      nose=1
      xsep=xsep/3.0

100     format(3(2x,E15.8))
```

```
444     format(8(2x,e15.8),2x,i3)
445     format(5(2x,e15.8),2x,'Xplane = ',i3)
446     format(1x'first row is x1-x5, second is y1-y5')

        Print*,'Enter slope for spline ends'
        read*, slope

        Print*,'Enter number of points for spline'
        read*, dim

         write(12,*) 'title=" "'
         write(12,*) 'variables=x,y,z'
         write(12,222) mcap*2-1+dim*2,ncap+dim-1
c        write(12,222) mcap*2-1+dim*2
 222       format(1x,'zone t=" ", i=',i4,', j=',i4,', f=point')
c222       format(1x,'zone t=" ", i=',i4,', f=point')

c
c
c       Set up to calculate the splines for the blunted nose
c
c
         z1=sep
         z2=z(2,mcap+1) + xsep
         z5=0.0
         y2=y(2,1)
         y1=y(1,1)
         y2=y(2,1)
         y4=y(2,mcap*2+1)
         y3=y(1,mcap*2+1)+(y4-y(1,mcap*2+1))/z2*z1
         x1=x(1,1)+(x(2,1)-x(1,1))/z2*z1
         x2=x(2,1)
         x5=x1-xsep
         y5=(y1+y3)*0.5
c
c
c  Write out the nose coordinates
c
c
        Do 35 i=1,mcap*2+dim*2-1
          write(12,100) x1,y5,z5
35       continue
c
```

```
c
c     Calculate the X-Y plane splines
c
c
      tau(1)=x5
      tau(2)=x1
      c(1,1)=y5
      c(1,2)=y1
      c(2,1)=-1.0*slope
      c(2,2)=0.0
      ncub=2
      ibc1=1

      call cubspl (tau,c,ncub,ibc1,ibc1)

      h = (x1-x5)/FLOAT(dim)

      Do 30 ii=1,dim
        xnose(ii) = x5 + h*FLOAT(ii)
        xt = h*FLOAT(ii)
        ynose(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &            /3.0)/2.0)
30     continue

      c(1,2)=y3
      c(2,1)=slope
      c(2,2)=(y3-y1)/(x2-x1)

      call cubspl (tau,c,ncub,ibc1,ibc1)

      Do 40 ii=1,dim
        xt = h*FLOAT(ii)
        ynose(ii+dim) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &            /3.0)/2.0)
40     continue
c
c
c     Do the X-Z Plane spline
c
c
      c(1,1)=z5
      c(1,2)=z1
      c(2,1)=2.0*slope
      c(2,2)=(z2-z1)/(x2-x1)

      call cubspl (tau,c,ncub,ibc1,ibc1)

      Do 50 ii=1,dim
        xt = h*FLOAT(ii)
        znose(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &            /3.0)/2.0)
50     continue

c
c
c     Now, do the linear interpolation to fill in the new
c     Y-Z planes with X coordinates given by xnose(i)
c     First, though, we need to get the spline fit for
c     the Y-Z plane at X(nose).
c
c
       nose = 2
        y1=y(nose,mcap+1) - sep
        y2=y(nose,mcap)
        y3=y(nose,mcap+1)
        y4=y(nose,mcap+2)
        y5=(y3+y1)*.5
        x1=z(nose,mcap+1)
        x2=z(nose,mcap)
        x3=z(nose,mcap+1)
        x4=z(nose,mcap+2)
        x5=x1+xsep
        tau(1) = x1
        tau(2) = x5
        c(1,1) = y1
        c(1,2) = y5
        c(2,1) = (y1-y2)/(x1-x2)
        c(2,2) = slope
        ncub = 2
        ibc1 = 1

        call cubspl (tau,c,ncub,ibc1,ibc1)

        h = (x5-x1)/FLOAT(dim)

        Do 60 ii=1,dim
```

```
            xsp(ii) = x1 + h*FLOAT(ii)
            xt = h*FLOAT(ii)
            ysp(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &                 /3.0)/2.0)
60       continue

          tau(1) = x3
          tau(2) = x5
          c(1,1) = y3
          c(1,2) = y5
          c(2,1) = (y3-y4)/(x3-x4)
          c(2,2) = -1.0*slope
          ncub = 2
          ibc1 = 1

          call cubspl (tau,c,ncub,ibc1,ibc1)

          h = (x5-x3)/FLOAT(dim)

          Do 70 ii=1,dim
            xsp(ii) = x3 + h*FLOAT(ii)
            xt = h*FLOAT(ii)
            yspcb(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &                 /3.0)/2.0)
70       continue
c
c
c       Now start the interpolation
c
c
        Do 80 j=2,dim-1
c          write(12,*) 'zone'
c
c
c  The freestream surface
c
c
          Do 90 i=1,mcap
            y1=(y5-ynose(j))/(y5-y(nose,1))
     &           *(y(nose,i)-y(nose,1)) + ynose(j)
            z1=(znose(j)/(z(nose,mcap+1)))*z(nose,i)
            write(12,100) xnose(j),y1,z1
90         continue
```

```
c
c
c  The freestream spline
c
c
       Do 110 i=1,dim-1
        y1=(y5-ynose(j))/(y5-y(nose,1))
   &           *(ysp(i)-y(nose,1)) + ynose(j)
        z1=(znose(j)/(z(nose,mcap+1)))*xsp(i)
        write(12,100) xnose(j),y1,z1
110     continue
c
c
c  The compression spline
c
c
       Do 120 i=dim-1,1,-1
        y1=ynose(j+dim)-(ynose(j+dim)-y5)/(y(nose,mcap*2+1)-y5)
   &           *(y(nose,mcap*2+1)-yspcb(i))
        z1=(znose(j)/(z(nose,mcap+1)))*xsp(i)
        write(12,100) xnose(j),y1,z1
120     continue
c
c
c  The compression surface
c
c
       Do 130 i=mcap+1,2*mcap+1
        y1=ynose(j+dim)-(ynose(j+dim)-y5)/(y(nose,mcap*2+1)-y5)
   &           *(y(nose,mcap*2+1)-y(nose,i))
        z1=(znose(j)/(z(nose,mcap+1)))*z(nose,i)
        write(12,100) xnose(j),y1,z1
130     continue
80     continue
c
c
c     Now do the main portion of the waverider
c
c
      write(6,*) 'Nose starts at plane',nose

      Do 140 mm=nose,ncap
```

```
c          write(12,*) 'zone'

         print*, 'Loop number',mm

         y1=y(mm,mcap+1) - sep
         y2=y(mm,mcap)
         y3=y(mm,mcap+1)
         y4=y(mm,mcap+2)
         y5=(y3+y1)*.5
         x1=z(mm,mcap+1)
         x2=z(mm,mcap)
         x3=z(mm,mcap+1)
         x4=z(mm,mcap+2)
         x5=x1+xsep
         tau(1) = x1
         tau(2) = x5
         c(1,1) = y1
         c(1,2) = y5
         c(2,1) = (y1-y2)/(x1-x2)
         c(2,2) = slope
         ncub = 2
         ibc1 = 1

         call cubspl (tau,c,ncub,ibc1,ibc1)

         h = (x5-x1)/FLOAT(dim)

         Do 150 ii=1,dim-1
           xsp(ii) = x1 + h*FLOAT(ii)
           xt = h*FLOAT(ii)
           ysp(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
      &                 /3.0)/2.0)
150      continue

         Do 160 ii=1,mcap
           write(12,100) x(mm,ii),y(mm,ii),z(mm,ii)
160        Continue

         Do 170 ii=1,dim-1
           write(12,100) x(mm,ii),ysp(ii),xsp(ii)
170        COntinue

         tau(1) = x3
```

```
        tau(2) = x5
        c(1,1) = y3
        c(1,2) = y5
        c(2,1) = (y3-y4)/(x3-x4)
        c(2,2) = -1.0*slope
        ncub = 2
        ibc1 = 1

        call cubspl (tau,c,ncub,ibc1,ibc1)

        h = (x5-x3)/FLOAT(dim)

        Do 180 ii=1,dim-1
          xsp(ii) = x3 + h*FLOAT(ii)
          xt = h*FLOAT(ii)
          ysp(ii) = c(1,1) + xt*(c(2,1)+xt*(c(3,1)+xt*c(4,1)
     &             /3.0)/2.0)
180     continue

        Do 190 ii=dim-1,1,-1
          write(12,100) x(mm,ii),ysp(ii),xsp(ii)
190     COntinue

         Do 200 ii=mcap+1,mcap*2+1
          write(12,100) x(mm,ii),y(mm,ii),z(mm,ii)
200     Continue
140     continue

111    format(1x,3(e15.8,2x),'Free Stream Surface')
112    format(1x,3(e15.8,2x),' Compression Surface')
113    format(5x,'X',17x,'Y',17x,'Z',17x,'Plane : ',I4)
c
     RETURN
       END
```

```
************************************************************************
      SUBROUTINE GEOM (sum,a1,num,astep)

      IMPLICIT REAL*8 (a-h,o-z)
      REAL*8 astep(num),xsum,feval,jac,dr,rinit,
     *                rdif,fstop,rmax

      rinit = 1.0d0
      fstop = 0.00010d0
      icount = 0
      imax = 10
      rmax = 1.5d0
      rdif = 0.01d0
      xsum = sum/a1
1      rinit = rinit + rdif
      dr = rinit
      icount = 0
      IF (rinit.gt.rmax) THEN
       WRITE (*,*) ' rinit exceeded rmax'
       GO TO 99
       END IF

10     CONTINUE
      feval = 1.0d0
      DO 20 i=1,num-1
20        feval = feval + dr**i
      feval = feval - xsum
      IF (ABS(feval).lt.fstop) GO TO 89
      jac = 1.0d0
      DO 30 i=1,num-2
30        jac = jac + (i+1)*dr**i
      dr = dr - feval/jac
      icount = icount + 1
      IF (icount.eq.imax) THEN
         GO TO 1
         ELSE
       GO TO 10
       END IF

89     astep(1) = a1
      r = dr
      DO 40 i=2,num
40       astep(i) = a1*r**(i-1)
```

```
99   RETURN
    END

************************************************************************
      SUBROUTINE CUBSPL (TAU, C, N, IBCBEG, IBCEND)
C     ************************* INPUT ****************************************
C      N=Number of data points.  Assumed to be >=2.
c     (tau(i), c(1,i), i=1,n)=Abcissa and ordinates of the
c     data points.  Tau is assumed to be strictly increasing.
c     IBCBEG, IBCEND = boundary condition indicators and
c     c(2,1), c(2,n) = boundary condition information, specifically,
c       IBCBEG=0 means no boundary condition at tau)1) is given.
c         In this case, the not-a-knot condition is used, i.e. the
c         jump in the third derivative across tau(2) is forced to
c         0, thus the first and second cubic polynomial pieces
c         are made to coincide.
c       IBCBEG=1 means that the slope at tau(1) is made to equal
c         to c(2,1), supplied by input.
c       IBCBEG=2 means that the second derivative at tau(1) is
c         made equal to c(2,1), supplied by input.
c       IBCEND=0, 1, or 2 has analogous meaning concerning the
c         boundary condition at tau(n), with the additional
c         information taken from c(2,n).
C     ************************* OUTPUT ********************************
c     c(j,i), j=1, 4; I=1, L (L=N-1) = the polynomial coeffiecients
c       of the cubic interpolating spline with the interior knots (or
c       joints) tau(2), tau(n-1).  Precisely, in the interval (tau(I),
c       tau(I+1)), the spline F is given by
c         F(x)=c(I,1)+h*(c(2,I)+h*(c(3,I)+h*c(4,I)/3)/2)
c       where h=x-tau(I).  The function program *PPVALU* may be
c       used to evaluate F or its derivatives from tau, c, L and K=4
C
      INTEGER IBCBEG, IBCEND,N,I,J,L,M
      REAL*8 C(4,N),TAU(N),DIVDF1,DIVDF3,DTAU,G
C
      L = N-1
C
      DO 10 M=2,N
        C(3,M) = TAU(M) - TAU(M-1)
        C(4,M) = (C(1,M) - C(1,M-1))/C(3,M)
10     CONTINUE
C
      IF (IBCBEG-1)   11,15,16
```

```
11      IF (N .GT.2)  GOTO 12
C
        C(4,1) = 1.0
        C(3,1) = 1.0
        C(2,1) = 2.0*C(4,2)
        GOTO 25
C
12      C(4,1) = C(3,3)
        C(3,1) = C(3,2) + C(3,3)
        C(2,1) = ((C(3,2)+2.0*C(3,1))*C(4,2)*C(3,3)+C(3,2)**2*C(4,3))
     &          /C(3,1)
        GOTO 19
C
15      C(4,1) = 1.0
        C(3,1) = 0.0
        GOTO 18
C
16      C(4,1) = 2.0
        C(3,1) = 1.0
        C(2,1) = 3.0*C(4,2)-C(3,2)/2.0*C(2,1)
18      IF (N .EQ. 2) GOTO 25
C
19      DO 20 M=2,L
         G = -C(3,M+1)/C(4,M-1)
         C(2,M) = G*C(2,M-1)+3.0*(C(3,M)*C(4,M+1)+C(3,M+1)*C(4,M+1))
         C(4,M) = G*C(3,M-1)+2.0*(C(3,M)+C(3,M+1))
20      CONTINUE
C
        IF (IBCEND -1) 21,30,24
21      IF (N .EQ. 3 .AND. IBCBEG .EQ. 0) GOTO 22
C
        G = C(3,N-1)+C(3,N)
        C(2,N) = ((C(3,N)+2.0*G)*C(4,N)*C(3,N-1)
     &          +C(3,N)**2*(C(1,N-1)-C(1,N-2))/C(3,N-1))/G
        G = -G/C(4,N-1)
        C(4,N) = C(3,N-1)
        GOTO 29
C
22      C(2,N) = 2.0*C(4,N)
        C(4,N) = 1.0
        GOTO 28
C
24      C(2,N) = 3.0*C(4,N)+C(3,N)/2.0*C(2,N)
```

```
      C(4,N) = 2.0
      GOTO 28
25     IF (IBCEND - 1)  26,30,24
26     IF (IBCBEG .GT. 0) GOTO 22
C
      C(2,N) = C(4,N)
      GOTO 30
28     G = -1.0/C(4,N-1)
C
29     C(4,N) = G*C(3,N-1) + C(4,N)
      C(2,N) = (G*C(2,N-1)+C(2,N))/C(4,N)
C
30     DO 40 J=L,1,-1
        C(2,J) = (C(2,J)-C(3,J)*C(2,J+1))/C(4,J)
40     CONTINUE
C
      DO 50 I=2,N
        DTAU = C(3,I)
        DIVDF1 = (C(1,I)-C(1,I-1))/DTAU
        DIVDF3 = C(2,I-1)+C(2,I)-2.0*DIVDF1
        C(3,I-1) = 2.0*(DIVDF1-C(2,I-1)-DIVDF3)/DTAU
        C(4,I-1) = (DIVDF3/DTAU)*(6.0/DTAU)
50     CONTINUE

      RETURN
      END
```

# APPENDIX D: CN1DAT CONTROL FILE

```
ICASE,IMTHDX,IMTHDY,IMTHDZ 1=Roe,2=McC,3=SW,4=vL

22   1     1     1
NEND
100
INS
1
IL, JL, KL
30, 47, 53
ILCTST,ICFL,CFLEXP,CFLMAX,CFL
1    5   1     0.9   0.01
IREST,CFCRHO,CFCEI,CFLPEN,CEXPPEN,INOFRZ
 1   0.1    0.1   1.    0.     5
(I/J/K)LMTR,OMEGA,OETA(I/J/K),R(I/J/K)DELT,  (I/J/K)ENTH,(I/J/K)ISO

2   2   2   1.    -1 -1 -1    0.05 0.05 0.05 2  2    2    0    1   1
R(I/J/K)SPR
0  0  0
IDD(1/2/3/4/5)
0 1 0 0 1
JDD(1/2/3/4/5)
0 0 1 0 1
KDD(1/2/3/4/5)
0 0 0 1 1
IPC,IMPLT,NSWPS,COEF,AA
2   1    2     0.   1.5
IADBWL,ALPHA,PHI,TWALL,RMINF,REL,        RLSCL,TINF,IMETRC

0   0.000000 0.  530.00 10.  1.0653E6    1.00   408.57 0
ITURB,ITBFRZ,APLUS,CCP,    CKLEB,CWK,        SMLK,BIGK,                    PRT,
CMUTM,ISIMTR
0    1      26.    2.08 0.3    0.25   0.4   1.68E-02   0.9   14.    0
IREAD,IGRID,IP3DOP,IDGBUG,MODPR,IP3DMD,NRST,IFMRTI,IFMRTO,IINT

0    2     2      0     10      500000 0     1       1       1
IEXRNGS
0
-----------------------------------
IF IEXRNGS NON ZERO THEN,FOR
IDMY=1,IEXRNGS
CHRDMY,
IEXMTHD,IIEXRL,IIEXRH,IJEXRL,IJEXRH,IKEXRL,IKEXRH,IEXLMT

----------------------------------------
JEXRNGS
0
----------------------------------
IF JEXRNGS NON ZERO THEN,
FOR JDMY=1,JEXRNGS
CHRDMY
JEXMTHD,JIEXRL,JIEXRH,JJEXRL,JJEXRH,JKEXRL,JKEXRH,JEXLMT

----------------------------------------
KEXRNGS
0
----------------------------------
IF KEXRNGS NON ZERO THEN,FOR
KDMY=1,KEXRNGS
CHRDMY
KEXMTHD,KIEXRL,KIEXRH,KJEXRL,KJEXRH,KKEXRL,KKEXRH,KEXLMT

----------------------------------------
BOUNDARY CONDITIONS: Convention
     IBCTYPE=1 ==> Freestream fixed boundary condition
             2 ==> Solid boundary
             3 ==> Symmetry boundary
             4 ==> Singular boundary
             5 ==> Zero gradient boundary conditon
             6 ==> Constant gradient boundary condition
             7 ==> None
             8 ==> Angle of attack (not mature)
             9 ==> Pie
            10 ==> Periodic
            11 ==> Characteristic boundary conditions
IFACORD(IFACE),IRNGS(IFACE)  Face 1
1 1
FOR EACH RNG
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC
1 1 1 47 1 53 1 0 0 0 0 0 0
Face 2
2 1
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

30 30 1 47 1 53 5 1 1 1 999 999 999
```

```
Face 3
3 2
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

1 3 1 1 1 53 9 1 1 1 0 0 1
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

3  30 1 1 1 53 2 1 1 1 999 999 999
Face 4
4 1
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

1 30 47 47 1 53 5 1 1 1 999 999 999
Face 5
5 1
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

1 30 1 47 1 1 3 1 1 -1 999 999 999
Face 6
6 1
IBC(ST/EN),JBC(ST/EN),KBC(ST/EN),IBCTYPE,(I/V/W)UBC,(I/J/K)INDBC

1 30 1 47 53 53 3 1 1 -1 999 999 999
```

# Bibliography

1. Anderson, Dale A. et al. *Computational Fluid Mechanics and Heat Transfer.* New York, Hemisphere Publishing, 1984.

2. Anderson, John D. *Fundamentals of Aerodynamics.* New York, McGraw-Hill, 1984.

3. Anderson, John D. *Hypersonic and High Temperature Gas Dynamics.* New York, McGraw-Hill, 1989.

4. Anderson, John D. et al. Hypersonic Waveriders for Planetary Atmospheres. *AIAA 90-0538 28th Aerspace Sciences Meeting and Exhibit.* Reno Nevada, January 1990.

5. Anderson, John D. et al. Hypersonic Waveriders for High Altitude Applications. *AIAA 91-0530 29th Aerspace Sciences Meeting and Exhibit.* Reno Nevada, January 1991.

6. Anderson, John D. et al. "Hypersonic Waveriders: Effects of Chemically Reacting Flow and Viscous Interaction", *AIAA 92-0302 30th Aerspace Sciences Meeting and Exhibit.* Reno Nevada, January 1992.

7. Anderson, John D. *Modern Compressible Flow With Historical Perspective.* New York, McGraw-Hill, 1982.

8. Beran, P. Class handout distributed in Aero 752, Computational Fluid Dynamics. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB Ohio, March 1992.

9. Beran P. Class handout distributed in Aero 753, Computational Fluid Dynamics. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB Ohio, July 1992.

10. Bowcutt, K.G. et al. *Viscous Optimized Hypersonic Waveriders,* AIAA 87-0272, January 1987.

11. Bowcutt, K.G. et al. "Numerical Optimization of Conical Flow Waverider Including Detailed Viscous Effects", *Aerodynamics of Hypersonic Lifting Vehicles,* AGARD-CPP-428. March 1987

12. Corda, Stephen. et al. *Viscous Optimized Waveriders,* AIAA 88-0369 26th Aerospace Sciences Meeting and Exhibit. Reno Nevada, January 1988.

13. Corda, Stephen. *Viscous Optimized Waveriders Designed from Flows over Cones and Minimum Drag Bodies.* PhD Dissertation, Department of Aerospace Engineering, University of Maryland, College Park Maryland, 1988.

14. de Boor, Carl. *A Practical Guide to Splines.* New York, Springer-Verlag, 1978.

15. Gaitonde, Datta. *Computation of Viscous Shock/Shock Hypersonic Interactions with an Implicit Flux Split Scheme. Final Report,* 1 September 1989-1 September 1990. Wright-Patterson AFB Ohio: Universal Energy Systems, December 1990, (WRDC-TR-90-3076.

16. Gaitonde, Datta. *The Performance of Flux-Split Algorithms in High-Speed Flows.* AIAA 92-0186 30th Aerospace Sciences Meeting and Exhibit. Reno Nevada, January 1992.

17. Gaitonde, Datta. Personal Interviews. WL/FIMM, Wright-Patterson AFB Ohio, 1 July through 30 October 1992.

18. Hasen, G. Class handout distributed in Aero 624, Advanced Hypersonics. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB Ohio, July 1992.

19. Hayes, Wallace D. and Probstein, Ronald F. *Hypersonic Flow Theory,* New York, Academic Press, 1959.

20. Küchemann, Dietrich. *The Aerodynamic Design of Aircraft.* New York, Pergamon Press, 1978.

21. McLauglhin, Thomas A. *Viscous Optimized Waveriders for Chemical Equilibrium Flow,* M.S. Thesis. Department of Aerospace Engineering, University of Maryland, College Park Maryland, 1990.

22. Müller, B. et al. "Simulation of Hypersonic Waverider Flow". *Proceeding of the 1st International Hypersonic Waverider Symposium.* University of Maryland, College Park Maryland, October 1990.

23. Murthy, T. K. S. *Computational Methods in Hypersonic Aerodynamics.* Computational Mechanics Publications, Kluwer Aerodynamics Publishers, 1992.

24. Nonweiler, Terence R.F. "The Waverider Wing In Retrospect and Prospect - A Personalized View", *Proceeding of the 1st International Hypersonic Waverider Symposium.* University of Maryland, College Park Maryland, October 1990.

25. Rasmussen, Maurice L. *Optimization of Waverider Configurations Generated from Axisymmetric Conical Flows.*

26. Rasmussen, Maurice L. et al. *On Waverider Shapes Applied to Aero-Space Plane Forebody Configurations.*

27. Rasmussen, Maurice L. Class Notes distributed in Aero 630, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio

28. Stecklein, Capt Gregory O. *A Comparative Study of Numerical Versus Analytical Waverider Solutions.* M.S. thesis, AFIT/ENY/GAE91D-26. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB Ohio, December 1991.

29. Steger, Joseph L. and Warming, R.F. "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods", *Journal of Computational Physics, 40:* 263-293 (April 1980)

30. Steinbrenner, John P. et al. *The GRIDGEN 3D Multiple Block Grid Generation System: Interim Report,* 1 October 1987-1 October 1990. Contract F33615-87-C-3003. Fort Worth Texas: General Dynamics Corporation, April 1991 (WRDC-TR-90-3022).

31. Takashima, N. et al. "Navier-Stokes Computation of a Viscous Optimized Waverider", *AIAA 92-0305 30th Aerspace Sciences Meeting and Exhibit.* Reno Nevada, January 1992.

32. Townend, L.H. "Research and Design for Lifting Reentry", *Progress in Aerospace Sciences.* 18:1-80, 1979.

33. Vanmol, D. *Aerodynamic Heating to Hypersonic Waveriders*, M.S. Thesis, Department of Aerospace Engineering, University of Maryland, College Park Maryland, 1991

34. Vincenti, Walter D. et al. *Introduction to Physical Gas Dynamics.* Malabar Florida, John Wiley & Sons, 1965.

35. Warming, R.F et al. *Math. Comp.* 29. 1975 1037

36. White, Frank M. *Viscous Fluid Flow* (Second Edition). New York, McGraw-Hill, 1991.

37. Yee, H.C. *A Class of High-Resolution Explicit and Implicit Shock-Capturing*

*Methods*. NASA Technical Memorandum 101088, Ames Research Center, California, February 1989.

## Vita

First Lieutenant James A. Mundy V was born on 24 July 1965 in Roanoke Virginia. He received his GED from the state of Massachusetts in 1986, and enlisted in the United States Air Force. He attended the University of Maryland, graduating with a Bachelor of Science degree in Aerospace Engineering in 1988. Upon graduation he attended Air Force Officer Candidate School and received a reserve commission in the USAF. He began as a test engineer for the 49th Test and Evaluation Squadron at Barksdale AFB, Louisianna, where he was involved in the operational teseting and evaluation of Strategic Air Command's airborne weapons systems. Lt Mundy served at the 49th Test Squadron until May of 1991, when he entered the School of Engineering, Air Force Institute of Technology.

| | |
|---|---|
| **REPORT DOCUMENTATION PAGE** | *Form Approved*<br>*OMB No. 0704-0188* |

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

| | | |
|---|---|---|
| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE<br>December 1992 | 3. REPORT TYPE AND DATES COVERED<br>Master's Thesis |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>The Effects of Viscosity on a Conically Derived Waverider | 5. FUNDING NUMBERS |
| 6. AUTHOR(S)<br>James A. Mundy, 1Lt, USAF | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Air Force Institute of Technology<br>Wright-Patterson AFB, Ohio 45433 | 8. PERFORMING ORGANIZATION REPORT NUMBER<br>AFIT/GAE/ENY/92D-23 |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>WL/FI<br>Wright-Patterson AFB, Ohio 45433 | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER |

11. SUPPLEMENTARY NOTES

| | |
|---|---|
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT<br>Unlimited Distribution | 12b. DISTRIBUTION CODE |

13. ABSTRACT *(Maximum 200 words)*

This study investigated the effects of the interaction between the viscous boundary layer and the shock wave produced by a Mach 10 inviscid optimized waverider. An implicit, Roe flux-splitting algorithm, developed by WL/FIMM, was used to solve the flow field. A validation for the inviscid version of the CFD algorithm was accomplished by comparing the numerical data produced by the CFD code to the analytic results derived by Rasmussen, and by comparison to results of the explicit version of the same Roe flux-splitting code. The computational results compared favorably. The inviscid case studied using the implicit code produced results identical, for all practical purposes, to those of the explicit code, though approximately twice as quickly. The results of the viscous flow case matched well with the results predicted by theory. The lift to drag ratio calculated, 5.74, is comparable to the results of other researchers.

| | |
|---|---|
| 14. SUBJECT TERMS<br>Inviscid Optimized Waverider, Navier-Stokes Solution, Viscous Flow, Hypersonic Vehicle | 15. NUMBER OF PAGES<br>158 |
| | 16. PRICE CODE |

| | | | |
|---|---|---|---|
| 17. SECURITY CLASSIFICATION OF REPORT<br>UNCLASSIFIED | 18. SECURITY CLASSIFICATION OF THIS PAGE<br>UNCLASSIFIED | 19. SECURITY CLASSIFICATION OF ABSTRACT<br>UNCLASSIFIED | 20. LIMITATION OF ABSTRACT<br>UL |